Diplomacy AI Development Environment Message Syntax

Version 0.1~~7~~8. Last updated ~~11th September~~ 7th March 2010~~2009~~.

The following document defines the language which will be used for communications between a client and server, and for negotiation between two clients, in the Diplomacy AI Development Environment project. Statements which use "will", "shall" or "must" are mandatory, and must be observed. Statements which use "may" or "can be" are recommended, but cannot or will not be enforced.

The syntax is split into a number of levels. Each level completely includes all previous levels. The levels are :

Level   0 : No Press
Level 10 : Peace and Alliances
Level 20 : Order proposals
Level 30 : Multipart Arrangements
Level 40 : Sharing out the Supply Centres
Level 50 : Nested Multipart Arrangements
Level 60 : Queries and Insistences
Level 70 : Requests for suggestions
Level 80 : Accusations
Level 90 : Future discussions
Level 100 : Conditionals
Level 110 : Puppets and Favours
Level 120 : Forwarding Press
Level 130 : Explanations

All Bots must implement the commands in all levels – they should not assume that they will never be playing in a game that is above the level they are designed for. Where a Bot receives a message that is above its intended level, there is a response it can use to indicate this.

The Diplomacy AI Development Environment Message Syntax is based on the structure and syntax of the DPP Language written by Danny Loeb. However, a significant number of changes have been made from the DPP Language in order to make it more powerful and more consistent.

Common terms

There are a number of terms which are used by a lot of commands. They are :

Power – can be one of **AUS, ENG, FRA, GER, ITA, RUS, TUR**. These may be changed for variants other than Standard - see the **MAP** token.

Province – can be one of **ADR, AEG, ALB, ANK, APU, ARM, BAL, BAR, BEL, BER, BLA, BOH, BRE, BUD, BUL, BUR, CLY, CON, DEN, EAS, ECH, EDI, FIN, GAL, GAS, GOB, GOL, GRE, HEL, HOL, ION, IRI, KIE, LON, LVN, LVP, MAO, MAR, MOS, MUN, NAF, NAO, NAP, NTH, NWG, NWY, PAR, PIC, PIE, POR, PRU, ROM, RUH, RUM, SER, SEV, SIL, SKA, SMY, SPA, STP, SWE, SYR, TRI, TUN, TUS, TYR, TYS, UKR, VEN, VIE, WAL, WAR, WES, YOR.** Alternative abbreviations must not be used. These may be changed for variants other than Standard - see the **MAP** token.

Coast – can be one of **NCS, ECS, SCS, WCS.**

Season and Phase – can be one of **SPR** (Spring Moves), **SUM** (Spring retreats), **FAL** (Fall Moves), **AUT** (Fall retreats), **WIN** (Adjustments).

Unit type – can be one of **AMY, FLT.**

Notation

In the text, anything in [square brackets] may be omitted.

Anything which is specified twice followed by a line of dots … may be repeated any number of times (including once, but not including not at all unless it is also in [square brackets].

Any (round brackets) are part of the syntax of the language.

<u>Internal Representation</u>

Internally, all tokens are represented as a 16-bit value. Details of the value for each token are provided in the Client-Server Protocol Document. Conversion between 16-bit values and text can be done. All text versions of the tokens are 3 characters long, alphabetic only, and not case-sensitive.

<u>Level 0 : No Press</u>

In this section, we just have the syntax required for getting a game up and running, ordering, retrieving results, and general game management.

<u>(i) Starting the game.</u>

NME ('name') ('version')

This should be sent by the client to the server as soon as it has connected. It contains the name and version of the client. Both **name** and **version** are text fields. They may any printable character. The Server will respond with **YES(NME('name') ('version'))**, or **REJ (NME ('name') ('version'))** if the game has already started.

The name and version are stored by the Server, for storing game records when the game ends. It is not possible for a client to request information about which client is playing each power.

OBS

This is sent by the client to the server in place of **NME**, to indicate that the connected client is actually just an observer, and doesn't wish to take part in the game. Observers are informed of the game progress as turns process, but do not take any part in the game. The server will respond with **YES (OBS)**.

IAM (power) (passcode)

This is sent by the client to the server in place of **NME**, to indicate that the client is rejoining the game following a loss of connection. **Passcode** is the passcode which was sent to the client by the server in the original **HLO** message at the start of the game. The server will reply with **YES (IAM (power) (passcode))** if the takeover is allowed, or **REJ (IAM (power) (passcode))** if the passcode is wrong, the game is not waiting for a replacement of the specified power, or the game has not started. Following a successful **IAM** message, a **MAP** and **HLO** message are NOT sent. The client is straight into the game where it left off. Note that there may have been messages which were lost as the connection broke. These messages will not be resent on reconnection.

Until a successful **NME**, **OBS** or **IAM** message has been received, the client will be treated the same as an observer, except that messages will not be sent to the client except in response to messages from the client (e.g. **ORD**, **NOW**, **SCO**, etc will not be sent). The one exception to this is **OFF**, which may be sent to a client before a successful **NME**, **OBS** or **IAM** message has been received.

MAP ('name')

Once the client has joined the game, the **MAP** token is sent to specify the map which is to be used, as soon as the **HLO** or **OBS** has been received and acknowledged. The name is unique to the map - so if the client has played a game on a map with this name before, then this will be the same map again. The Standard map is called 'standard'. **name** can include only letters, numbers and underscore. Map names are not case sensitive.

The client may request another copy of the **MAP** message at any time, by sending **MAP** with no parameters.

The client should respond with **YES (MAP ('name'))** if it is to play on the map, or **REJ (MAP ('name'))** if the client can not cope with the requested map.

If the client does not know the map, then instead of replying with **YES (MAP ('name'))** or **REJ (MAP ('name'))**, it can request the map definition with :

MDF

The server will reply with the map definition of the form :

MDF (powers) (provinces) (adjacencies)

(powers) is the list of powers. e.g. **(AUS ENG FRA GER ITA RUS TUR)**.

(provinces) is the list of provinces. This is broken further into two sections - **((supply centres) (non supply centres))**.
(supply centres) is of the form **((power centre centre …) (power centre centre …) …)** where **centre** is a supply centre province, and **power** is the power which the centre is a home centre for. **power** is **UNO** for neutral supply centres. Also, in the unusual case of a variant where a supply centre is a home centre for two or more powers, **power** may be of the form **(power power …)**. Note that the home centres are the locations where a power may build new units, but are not necessarily the location where the units start the game.
(non supply centres) is the list of all non-supply-centre provinces.

(adjacencies) is the list of what is next to what. It is of the form **((prov_adj) (prov_adj) ...)**.

(prov_adj) is of the form **(province (unit_type adj_prov adj_prov ...) (unit_type adj_prov adj_prov ...) ...)**.

unit_type is one of :

AMY - list of provinces an army can move to
FLT - list of provinces a fleet can move to
(FLT coast) - list of provinces a fleet can move to from the given coast.

adj_prov is one of :

province - a province which can be moved to
(province coast) - a coast of a province that can be moved to.

The type of a province can be determined from its adjacencies. An inland province only has army adjacencies. A sea province only has fleet adjacencies. A coastal province has both. A multi-coastal province has entries for each coast.

Note that for a land province surrounded by sea, there will be an **AMY** entrance with no provinces listed.

So, the map for Standard would (with some provinces skipped, and carriage returns added for clarity) :

MDF (AUS ENG FRA GER ITA RUS TUR)
(((AUS TRI BUD VIE) (ENG LON LVP EDI) ... (UNO POR SPA TUN BEL HOL ...)) (ADR AEG ALB APU ...))
((ANK (FLT ARM BLA CON) (AMY CON SMY ARM))
(ADR (FLT ION ALB TRI VEN APU))
(CON (AMY BUL SMY ANK) (FLT BLA (BUL ECS) (BUL SCS) AEG SMY ANK))
(BUL (AMY GRE SER RUM CON) ((FLT ECS) RUM BLA CON) ((FLT SCS) CON AEG GRE))
(BOH (AMY TYR MUN SIL GAL VIE))
…)

No map will ever contain a province abbreviation which is also has a defined meaning in the Diplomacy AI Development Environment syntax, except that the tokens for provinces and powers on the Standard map may be reused on variant maps.

No map will ever contain more than 256 provinces.

Once the client has received the Map Definition, it should then reply with **YES (MAP ('name'))** if it is able to use the map, or **REJ (MAP ('name'))** if it is not. If the client wishes to do some processing on the map before the game starts, then it should do this before sending **YES (MAP ('name'))**.

The client can send the **NOW** and **SCO** commands to the Server to determine the supply centre ownership and starting positions for the variant. See below for details.

Note that for some variants, the map name may be different, but the actual map is the same as another variant (for example, for Fleet Rome). When this occurs, it can not be assumed that the supply centre ownership and starting positions will be the same as for the other variants which use the same (but differently named) map.

On all maps, the condition for a solo is having more than half the supply centres in the game. So for a map which has 34 or 35 centres, the victory condition is 18 centres.

HLO (power) (passcode) (variant)

The Server sends this to each client when the game starts. **power** is the power to be played by this client, or UNO for an observer. **passcode** is an integer value which is required to rejoin the game. **variant** is of the form **(variant option) (variant option)…**, and contains the details of any options for the game.

e.g. **HLO (ENG) (1234) ((LVL 20) (MTL 1200))**

This client is England. It's passcode is 1234. This is a level 20 game, with 20 minutes (1200 seconds) per movement turn.

variant option can be any of the following :

LVL n – game is of syntax level n. This variant option will always be specified.

MTL seconds – maximum number of seconds available for movement turns. If not specified, there is no time limit.

RTL seconds – maximum number of seconds available for retreat turns. If not specified, there is no time limit.

BTL seconds – maximum number of seconds available for build turns. If not specified, there is no time limit.

DSD - deadline stops on disconnection. If a client disconnects when it has orders due, the deadline timer stops.

AOA – Any orders accepted. Any orders which match the syntax for an order, and are for a unit owned by the client sending the order, will be accepted, whether they are legal or not. If not specified, then only orders which could be valid will be accepted. In an AOA game, any token which would normally be given as a THX note can instead be used as an ORD result.

Further variant options are added in Level 10.

The variant options will always be specified in the order they appear in this document.

The client can send the command **HLO** to the server with no arguments at any time. The server will reply by sending another copy of the **HLO** message, or **REJ(HLO)** if the game hasn't started.

(ii) Current Position

SCO (power centre centre…) (power centre centre…) …

This is sent from the Server to each client at the start of the game, and after each Fall Retreat turn (or Fall Movement turn if there are no retreats). It indicates the current supply centre ownership. Unowned centres are listed against the power name **UNO**. E.g. at the start of the game :

SCO (AUS BUD TRI VIE) (ENG LPL EDI LON) (FRA BRE MAR PAR) (GER KIE BER MUN) (ITA ROM NAP VEN) (RUS STP MOS WAR SEV) (TUR ANK CON SMY) (UNO NWY SWE DEN HOL BEL SPA POR TUN GRE SER RUM BUL)

The client can send the command **SCO** to the server with no arguments at any time. The server will reply by sending another copy of the latest **SCO** message, or the starting supply centre ownership if the game hasn't started.

NOW (turn) (unit) (unit) …

This is sent from the server to the client at the start of the game, and after every turn. It indicates the current turn, and the current unit positions. E.g. at the start of the game :

NOW (SPR 1901) (AUS FLT TRI) (AUS AMY BUD) (AUS AMY VIE) (ENG FLT LON) …

Units in bicoastal provinces have a province and coast bracketed together. E.g. :

(RUS FLT (STP SCS))

Before a retreat turn, units may have a list of retreat options, prefixed by **MRT** (Must retreat to). E.g.

(ENG FLT NTH MRT (LON YOR EDI NWG))

If the unit has no possible retreats, then the unit will still be listed, and the client still must order the disband. E.g.

(ENG FLT NTH MRT ())

Retreat options will include a coast if a fleet can retreat to a multi-coastal province. e.g.

(TUR FLT CON MRT (BLA SMY (BUL ECS) (BUL SCS)))

The client can send the command **NOW** to the server with no arguments at any time. The server will reply by sending another copy of the latest **NOW** message, or the starting position if the game hasn't started.

HST (turn)

Any previous turn's results may be requested by sending an **HST** command to the server with the turn as a parameter (e.g. **HST(SPR 1901)**). The server will reply by sending the **ORD** messages for that phase, followed by **SCO** and **NOW** messages to give the position after the turn requested. **SCO** will be sent after every **HST** command, not just those where supply centre ownership changes. **REJ (HST (turn))** is sent if the requested turn doesn't exist.

(iii) Submitting orders

SUB (order) (order) …
SUB (turn) (order) (order) ...

This is sent from the client to the server, to submit orders for the next turn. The second form will check that the turn stated is the current turn (i.e. the turn sent in the most recent NOW message), and will reply with REJ (SUB (turn) (order) (order) ...) if this is not the case

order can take the following form :

Movement phase orders.

(unit) HLD	Hold
(unit) MTO province	Move
(unit) SUP (unit)	Support to hold
(unit) SUP (unit) MTO prov_no_coast	Support to move
(unit) CVY (unit) CTO province	Convoy
(unit) CTO province VIA (sea_province sea_province …)	Move by convoy

Retreat phase orders

(unit) RTO province	Retreat to
(unit) DSB	Retreat off the board

Build phase orders

(unit) BLD	Build unit
(unit) REM	Remove unit
power WVE	Waive build

unit is a unit, and must be in the form as given by the **NOW** command – i.e. **power type province** – e.g. **ENG FLT LON**. Where a unit is a fleet in a multi-coasted province, the coast must be specified.
province is any province. For a fleet moving to a province with multiple coasts (Spain, StP or Bul in Standard), it must be of the form **(province coast)**.
prov_no_coast is a province with no coast specification.
sea_province is a province where a fleet may be to convoy. The provinces must be listed in the order that the army will pass through them.

Examples:

SUB ((ENG AMY LVP) HLD) ((ENG FLT LON) MTO NTH) ((ENG FLT EDI) SUP (ENG FLT LON) MTO NTH)

SUB ((ENG FLT BAR) MTO (STP NCS)) ((ENG FLT NWY) SUP (ENG FLT BAR) MTO STP)

SUB ((ITA AMY TUN) CTO SYR VIA (ION EMS)) ((ITA FLT ION) CVY (ITA AMY TUN) CTO SYR) ((ITA FLT EMS) CVY (ITA AMY TUN) CTO SYR)

SUB ((ENG AMY YOR) CTO NWY VIA (NTH)) ((ENG FLT NTH) CVY (ENG AMY YOR) CTO NWY) ((ENG FLT NWG) SUP (ENG AMY YOR) MTO NWY)

SUB ((ITA FLT NAP) BLD)

SUB ((RUS FLT GOB) REM)

All unused builds must be waived, even if they are unusable.

The Server will reply to each order (not each message), in left to right order through the orders, with a **THX** message. This is of the format :

THX (order) (note)

order is the order as submitted. **note** is one of the following :

MBV	Order is OK.
FAR	Not adjacent.
NSP	No such province
NSU	No such unit
NAS	Not at sea (for a convoying fleet)
NSF	No such fleet (in **VIA** section of **CTO** or the unit performing a **CVY**)
NSA	No such army (for unit being ordered to **CTO** or for unit being **CVY**ed)
NYU	Not your unit
NRN	No retreat needed for this unit
NVR	Not a valid retreat space
YSC	Not your supply centre
ESC	Not an empty supply centre
HSC	Not a home supply centre
NSC	Not a supply centre
CST	No coast specified for fleet build in StP, or an attempt to build a fleet inland, or an army at sea.
NMB	No more builds allowed
NMR	No more removals allowed
NRS	Not the right season

Alternatively it will reply with **REJ (SUB (order) (order) ...)** if the game has not started.

Any order which receives a note other than **MBV** has been rejected by the server, and a corrected order should be submitted in its place.

Movement and retreat orders can be changed by giving a new order for a unit. Builds and retreats can be cancelled by ordering **NOT (SUB (order))**. The Server will reply to this with **YES (NOT (SUB (order)))**, or **REJ (NOT (SUB (order)))** if the order cannot be cancelled. Alternatively, an entire turns orders can be cleared with **NOT (SUB)**, which the Server will reply to with **YES (NOT (SUB))**.

MIS (unit) (unit) …
MIS (unit MRT (province list)) (unit MRT (province list)) …
MIS (number)

This command can be sent by the Server immediately after the last **THX** message following a **SUB** command. If sent, it indicates that it does not have a full set of orders. The first form is used during a movement phase, and indicates the list of units which have not been ordered. The second form is used during a retreat phase, and indicates the list of dislodged units which have not been ordered. The third form is used during a build phase. If **number** is positive, it indicates that the client must order that many more disbands. If **number** is negative, it indicates that the client must order that many more builds.

The first two forms of the **MIS** command use exactly the same parameter format as the **NOW** command.

The client may request a copy of the current **MIS** command by sending **MIS** with no parameters. The server will respond with **REJ (MIS)** if the game has not started or has ended. If there are no outstanding orders, then the Server will reply with **MIS** with no parameters.

NOT(GOF)

This is sent by the client to the server. It means don't process orders until the deadline. It is the equivalent of set wait on the judges. The server will reply with **YES(NOT(GOF))**, or **REJ(NOT(GOF))** if the game has not started, the power has been eliminated, or the game has ended..

GOF

Go ahead and process orders when everybody is in. Server will reply with **REJ(GOF)** if the game has not started or has ended, or **YES(GOF)** if everything is OK. It will immediately follow this with **MIS(…)** (as when an incomplete set of orders is submitted – see **SUB**) if not all orders are in for this player.

If the client does not send **GOF** or **NOT(GOF)** during the turn, the server will assume the client is ready to process as soon as the client has submitted a complete set or orders (i.e. **GOF** is assumed each turn).

(iv) Results

ORD (turn) (order) (result)

This is sent by the Server when the turn has processed. **turn** is the turn which has just processed.

order is an order submitted by a player. One **ORD** message is sent per unit for a movement phase, one per retreat for a retreat phase, and one per build/disband/waive for a build phase . It has exactly the same format as in the **SUB** and **THX** commands.

result is the result of the order. It can be one of the following :

SUC	Order succeeded (can apply to any order).
BNC	Move bounced (only for MTO, CTO or RTO orders).
CUT	Support cut (only for SUP orders).
DSR	Move via convoy failed due to dislodged convoying fleet (only for CTO orders).
NSO	No such order (only for SUP, CVY or CTO orders).

In addition, a second token may be included :

RET   Unit was dislodged and must retreat.

In the case of a unit which was convoying or holding, **RET** may be the only token.

Retreat orders will always have a **result** of **SUC** or **BNC**. **BNC** indicates the unit was destroyed by a bounced retreat.

Build orders will always have a **result** of **SUC**.

Waived builds are reported as **(power WVE)** for the order. E.g. :

ORD (WIN 1901) (RUS WVE) (SUC)

If a power waives multiple builds, then multiple waive **ORD** messages are sent.

When the turn processes, the series of **ORD** messages will be immediately followed by a **SCO** (if the centre ownership has just been updated) and a **NOW** message. The **NOW** message will always be the last message in this sequence.

Turns may be skipped (e.g. a retreat turn where nobody has any retreats). The **NOW** message indicates which is the next turn.

The previous turn's results may be requested by sending an **ORD** command to the server with no parameters. The server will reply by resending the last movement turn's **ORD** message, and any subsequent Retreat or Build **ORD** messages, or **REJ(ORD)** if the game has not started or the first turn has not yet processed.

(v) Saving and Loading a game

SVE ('gamename')

The Server can send this at any point. It is a request for the client to save the current game with the name given. **gamename** will never be more than 8 characters, and can include only letters, numbers and underscore. Game names are not case sensitive. The client should save everything it will need in order to be able to resume the game at a later date. Once the game is saved, the client

should respond with **YES (SVE ('gamename'))**. If there is an error saving the game, then it should respond with **REJ (SVE ('gamename'))**. Once the game has been saved, the client should continue playing.

LOD ('gamename')

The server can send this to the client instead of sending a **HLO** message. It indicates that the game specified is to be loaded and restarted, rather than a new game started. See **SVE** for details of **gamename**. The client should load the game and then respond with an **IAM** command (as detailed above), and then start playing from the loaded position. If the game can not be reloaded (e.g. because a game with that name has never been saved), then it should respond with **REJ (LOD ('gamename'))**.

If an client fails to reload the game, then the game will be started and the client in question will immediately be placed in CD.

Note that the client does not need to store the current game position, as it can immediately send the commands **HLO**, **NOW**, **SCO**, and **ORD** to find out the state of the game. All that needs to be stored, is the power being played, the passcode as provided by the **HLO** command, and details of past press and alliances that the client wishes to store.

OFF

This command is sent by the Server, and indicates that the client should exit. The client should exit without replying.

(vi) Deadlines

TME (seconds)

This message is sent by the server, in games where there is a time limit for the turn. It indicates the number of seconds until the next deadline. It is sent immediately after each turn has processed (i.e. after the **ORD**, **SCO** (where appropriate)**,** and **NOW** messages).

In addition, the client can request **TME** to be sent at other times. To do this, the client sends **TME (seconds)** to the server. The server responds with **YES(TME (seconds))** to confirm this, or **REJ(TME (seconds))** if **seconds** is negative, or is longer than the length of the longest deadline, or if there are no deadlines. The server will then send **TME (seconds)** to the client, **seconds** seconds before the deadline, every turn.

Having requested a **TME** message, the client can cancel it with the message **NOT(TME(seconds))**. The server will respond with **YES(NOT(TME(seconds)))** to confirm it has been cancelled, or **REJ(NOT(TME(seconds)))** if there was no such time request. Alternatively the client can send **NOT(TME)** to request that all **TME** requests are cancelled. This will always be responded to with **YES(NOT(TME))**.

Also, the client can send the message **TME** with no parameters. The Server will respond with **TME (seconds)** where **seconds** is the time to the next deadline, or **REJ (TME)** if there are no deadlines.

The client should not acknowledge the **TME** message from the server.

(vii) Errors and Failures

PRN (message)

This message is sent from the server to the client, and indicates that **message** was received from the client by the server, but does not have a correct set of parentheses. The **PRN** message will also not have matching parentheses. The client should cope with this. No reply should be sent to the server.

HUH (message)

This message is sent from the server to the client, and means that the server determined that **message** had a syntax error in it. The token **ERR** is inserted into the message immediately before the first offending token.

This error can also be caused by trying to use a form of the Diplomacy AI Development Environment syntax which is not available at the syntax level for the current game.

The **PRN** and **HUH** messages may also be sent from the client to the server, if the AI believes that a message from the server is incorrectly bracketed, or contains illegal syntax. However, the server and the client must not send **HUH** or **PRN** in response to a **HUH** or **PRN** message.

CCD (power)

This message is sent from the server to the client, and indicates that the specified power has been declared to be in Civil Disorder. This will normally happen because either the power failed to get its orders in by the deadline, or because the TCP/IP connection to the client has been broken. If the power returns, the server will send the command **NOT(CCD(power))**

If the game is of the **DSD** variant, then when a power disconnects which has not yet submitted orders, then server sends **NOT(TME(seconds))** immediately after the **CCD** message to indicate that the deadline timer has stopped, and sends **TME(seconds)** following the **NOT(CCD(power))** message when the timer restarts.

(viii) Admin Messages

ADM ('name') ('message')

This message is sent from the Client to the Server, and contains an admin message - i.e. a message to do with the running of the game (e.g. I'm having connection problems, or Server going down, or I have to leave shortly). It can be sent by any client - player or observer, and can be sent before, during or after the game. It should not be used for negotiation. The Server will forward the message to every client. The Server may be configured to refuse Admin messages, in which case it will reply to the Client with **REJ (ADM ('name') ('message'))**.

(ix) End of the game

SLO (power)

This command is sent from the server to the client, and indicates that the game has ended due to a solo by the specified power. It is sent after the SCO message and before the NOW message.

DRW

This command is sent by the client to the server, to indicate that the client would accept a DIAS draw at this point. The server responds with **YES(DRW)** , or **REJ(DRW)** if the game has not started or has ended, or the message is received from an eliminated power or an observer..

DRW can also be sent from the server to the client, to indicate that a draw has been declared. This will happen as soon as all non-eliminated clients have simultaneously indicated that they will accept a draw at that point.

If a turn processes after **DRW** is sent from the client to the server, then the command is ignored. The client must resend the command each turn it continues to want a draw.

The **DRW** command is extended in level 10.

NOT(DRW)

This command cancels any **DRW** message which has been sent previously by this client. The server will respond with **YES(NOT(DRW))**, except if the game has not started, has finished, the power is eliminated, or is an observer in the game, when it will respond with **REJ(NOT(DRW))**.

SMR (turn) (power ('name') ('version') centres [year_of_elimination]) (power ('name') ('version') centres [year_of_elimination]) …

The message is sent from the server to the client immediately after the **SLO** or **DRW** message, and informs the client as to who was playing each power. **centres** is the number of centres held by the power at the end of the game. If this is 0 then **year_of_elimination** is included, and is the year that the power dropped to 0 centres.

Level 10 : Peace and Allies

In this level, we add some very basic press between powers.

(i) Extending the HLO command.

In addition to the previous **variant options**, the following may also be given at the start of the game :

PDA – Partial draws are allowed (i.e. NoDIAS).

NPR – No press during retreat phases

NPB – No press during build phases

PTL seconds – number of seconds before a movement deadline that all press between powers must stop. If not specified, then 0.

<u>**(ii) Extending the DRW command**</u>

In a **PDA** variant game, the client can send the **DRW** command with a parameter – the list of powers in the draw :

DRW (power power power …)

If all surviving powers submit a **DRW** command with the same list of powers on the same turn, then a draw is immediately declared including those powers.

Multiple **DRW** commands may be submitted, each one does <u>not</u> supersede previous **DRW** messages. Instead it adds an additional draw combination which the power will accept.

The Server will respond to each **DRW** message with **YES(DRW(power power power …))**, or **REJ(DRW(power power power …))** if the game has not started, or if the list of powers includes an eliminated power.

When a draw is declared in a **PDA** variant game, the draw message includes a parameter – the list of powers in the agreed draw.

The client may still send a **DRW** command without a parameter in a **PDA** game – it is the same as sending a **DRW** command listing all surviving powers.

The client can cancel a partial draw request with **NOT (DRW (power power power …))**. Sending **NOT(DRW)** cancels a **DRW** sent with no parameters, but does not cancel a DRW sent with a list of powers.

<u>(iii) **Sending and Receiving Press**</u>

SND (power power …) (press_message)
SND (power power …) (reply)
SND (turn) (power power …) (press_message)
SND (turn) (power power …) (reply)

These are sent by the client to the server, to send **press_message** to the list of **country**s given. The recipient list must not include the sending power.

For the third and fourth forms, the server will check that turn is the current turn (i.e. the turn sent in the most recent NOW message), and will reply with REJ(SND (turn) (power power …) (press_message)) or REJ (SND (turn) (power power …) (reply)) if this is not the case.

The message sent applies to the current turn, unless indicated otherwise by the contents of the message.

The server will reply with one of the following :

CCD (power)

power is in CD, and so can not receive press. If multiple powers are listed in the **SND** command, and the client receives a **CCD** reply, then the message has not been sent to anybody (including the ones that are not in civil disorder).

OUT (power)

power has been eliminated from the game. If multiple powers are listed in the **SND** command, and the client receives an **OUT** reply, then the message has not been sent to anybody (including the ones that have not been eliminated).

HUH (SND (power power …) (press_message))
HUH (SND (turn) (power power …) (press_message))

Syntax of the **SND** message is illegal. This may be a genuine syntax error, or it may be because **press_message** contains a token which is not allowed at the language level of the game. **ERR** is inserted immediately before the first token to cause an error. See **HUH** in Level 0 for details.

REJ (SND (power power …) (press_message))
REJ (SND (turn) (power power …) (press_message))

Either the game has not started, or it is a retreat phase in a **NPR** variant game, or a build phase in a **NPB** variant game, is past the press deadline in a **PTL** variant game, or the sending power has been eliminated or is in the recipients list.

YES (SND (power power …) (press_message))
YES (SND (turn) (power power …) (press_message))

The message was sent successfully.

When the message is sent successfully, the target powers will receive the message in one of the following forms :

FRM (power) (power power …) (press_message)

FRM (power) (power power …) (reply)

The first **power** is the power which sent the message. The **power** list and the **press_message** are as in the **SND** command. Note that if the **SND** message contained a turn parameter, this isn't included in the **FRM** message.

We have now covered every command in the Diplomacy AI Development Environment syntax. The remainder of this document covers the contents of **press_message** and **reply**.

(iv) Press fundamentals

Sequences of press messages can be split into two categories. They can be attempting to form an agreement between a number of powers, or they can be providing or requesting information.

Agreements are formed by one power proposing an arrangement to a number of other powers (either using the **PRP** (level 10) or the **INS** (level 60) tokens). For the arrangement to be agreed, each of the receiving powers must reply to the sending power plus all of the other receiving powers, accepting the proposal (using the **YES** (level 10) token).

Most arrangements apply to a single turn. Where this is not the case, it is indicated in the description of the arrangement. If a proposal is made in one turn, but accepted in a subsequent turn, then the turn or turns it applies to are calculated based on the time at which the original proposal was sent, not the time at which it was agreed. Powers should not attempt to form an agreement over a proposal which is wholly in the past, or which has already been breached.

In some cases, the order in which a number of press messages occurred is important. Where this is the case, each power should consider the order in which it received the confirmations of its own messages, and the messages from the other powers, from the server. It is guaranteed that this order will be the same for all powers.

Where a number of agreements are made which conflict with each other, the latter agreement takes precedence. This has a number of effects, including the ability to unmake an agreement by making a conflicting agreement.

(iv) Proposing an ~~Offer~~Arrangement

press_message = PRP (arrangement)

This indicates that the sending power is proposing an arrangement. Some offers represent specific actions, while others (e.g. PCE) represent concepts, and as such, only have a vague definition. **arrangement** can be in one of the following forms :

arrangement = PCE (power power power …)

Arrange Peace between the listed powers. Eliminated powers must not be included in the power list. The arrangement is continuous (i.e. it isn't just for the current turn).

arrangement = ALY (power power …) VSS (power power …)

Arrange an Alliance between the powers in the first list. The second list is the powers to ally against. Eliminated powers must not be included in either power list. The arrangement is continuous (i.e. it isn't just for the current turn).

arrangement = DRW

Arrange a Draw. In the case of a **PDA** game, this may also be **DRW (power power power …)**. The list of powers must include at least two powers. Eliminated powers must not be included in the power list.

arrangement = SLO (power)

Propose a Solo to the specified power. Note that the client can't actually order a solo – but may be able to order its units in a way that causes one to occur.

arrangement = NOT (arrangement)

All of the available arrangements can have **NOT** placed in front of them to mean the opposite. Double negatives should not be used !

arrangement = NAR (arrangement)

All of the available arrangements can also have **NAR** placed in front of them. This represents the lack of an arrangement. To use the level 20 token, **XDO**, as an example, **XDO (order)** is an arrangement that **order** will be ordered. **NOT (XDO (order))** is an arrangement that **order** will not be ordered. **NAR (XDO (order))** is an arrangement that allows **order** to be ordered, or not.

NAR is most commonly used in order to undo a previous arrangement.

(vi) Responding to a message

To respond to a message, one of the following can be sent :

reply = YES (press_message)

Accept the arrangement

reply = REJ (press_message)

Reject the arrangement

reply = BWX (press_message)

Refuse to answer. None of the sending power's business.

reply = HUH (press_message)

The message proposal contains a token sequence which is too complicated for this client to understand – usually because the client is not able to handle all the tokens that are available for press at the current level. In this case, the token **ERR** should be inserted immediately before the first token which could not be understood.

The above responses should only be sent as a response to a received message. **press_message** should be identical to the received message.

(vii) Cancelling a proposal

press_message = CCL (press_message)

A press message which proposes an arrangement can be cancelled before the arrangement has been agreed, using a **CCL** message. This can also be used to cancel a **YES** response to a proposed arrangement, before all parties involved in the proposal have replied.

A **CCL** message which is sent after the arrangement has been agreed has no effect, and **CCL** may not be used to cancel a message which is not attempting to form an agreement.

(viii) Making a statement.

arrangement is true.

This is generally used either to pass information, or to unilaterally override a previously made agreement. There are no defined replies to **FCT**, although of course, a **FCT** message may cause other statements or proposals to follow.

(~~v~~ix) Reporting your abilities

If the client sends a response of **HUH**, it should immediately follow it with a second message informing the other client as to what it can understand. This is done with :

press_message = TRY (tokens)

List all of the tokens which the client can process when received in the **press_message** part of a **FRM** message. **Tokens** should be a subset of **PRP PCE ALY VSS DRW SLO NOT YES REJ BWX.** In addition, the following tokens from higher levels may also be included if they can be processed by the client : **XDO DMZ AND ORR SCD OCC INS QRY THK FCT IDK SUG WHT HOW EXP SRY FOR IFF THN ELS XOY YDO FRM FWD SND**

The client does not need to list **TRY** or **HUH** in **tokens**, and should not send a **HUH** message in response to a **TRY** or **HUH** message.

A client which can not process Press (i.e. it is only designed to work at Level 0) should respond to any **FRM** message with :

SND (power) (HUH (ERR press_message))
SND (power) (TRY ())

Where **power** is the sending power, and **press_message** is the received message.

All tokens which can be processed by the client may be included in games of any level (so that the client can just have a fixed **TRY** message which the client sends if it doesn't understand something). There is no requirement to remove higher level tokens from the **TRY** message in lower level games. The Server will remove all higher level tokens from the **TRY** message before passing it onto the receiving power, and may also reorder the tokens.

Level 20 : Order Proposals

In level 20, we add two new forms for **arrangement**.

arrangement = XDO (order)

This is a arrangement for the given order to be ordered. **order** uses the same format as in the **SUB** command.

An **XDO** arrangement applies to the next turn in which the order type is valid – so the next movement turn for a **HLD, MTO, SUP, CVY** or **CTO** order, the next retreat turn for a **RTO** or **DSB** order, and the next adjustment turn for a **BLD, REM** or **WVE** order.

arrangement = DMZ (power power …) (province province …)

This is an arrangement for the listed powers to remove all units from, and not order to, support to, convoy to, retreat to, or build any units in any of the list of provinces. Eliminated powers must not be included in the power list. The arrangement is continuous (i.e. it isn't just for the current turn).

Both of these are responded to using the responses in level 10.

Level 30 : Multipart Arrangements

Level 30 adds the ability to combine multiple arrangements.

arrangement = AND (arrangement) (arrangement) (arrangement) …

These arrangements are grouped, and should be accepted or rejected as a group. This is usually used for arrangements such as "I support you and you support me".

arrangement = ORR (arrangement) (arrangement) (arrangement) …

These alternative arrangements are grouped. The client should accept or reject them as a group, and then use further press (if necessary) to decide which part of the arrangement is to actually occur. This is usually used for offers such as "I will make one of the following moves".

ORR is logical inclusive-or. That is, doing more than one of the arrangements is not against the overall arrangements.

At level 30, **AND** and **ORR** must not be nested – they must not both appear in the same message, and neither of them is allowed to appear more than once. All **AND** and **ORR** offers must include at least two sub-offers. E.g. **AND (arrangement)** is not allowed.

Level 40 : Sharing out the Supply Centres

Level 40 adds the ability to discuss who will gain which supply centres and provinces.

arrangement = SCD (power centre centre …) (power centre centre …) …

The given supply centre distribution is arranged. This is typically used when an alliance is sharing out the centres of the powers they are allied against. Any **centre**s not listed are not part of the division. Eliminated powers must not be included in the distribution of centres. The offer is a long term plan, it doesn't necessarily indicate that the centres will be taken this turn.

arrangement = OCC (unit) (unit) …

This is an arrangement for a power to place a particular unit in a particular location. **unit** is in the same format as for **NOW**. For instance **PRP (OCC (ENG FLT BEL))**. Alternatively, the token **UNT** may be used to represent any unit type. E.g. **PRP (OCC (ENG UNT BEL))**. The offer is a long term plan, it doesn't necessarily indicate that the unit will be placed there this turn.

Level 50 : Nested Multipart Arrangements

Level 50 allows **AND** and **ORR** to be nested. So the client can do the following :

PRP (ORR ((FRA AMY PAR) MTO PIC) (AND ((FRA AMY PAR) MTO BUR) ((GER AMY MUN) MTO BUR)))

It also adds a new token. **CHO**.

arrangement = CHO (minimum maximum) (arrangement) (arrangement) (arrangement) ...

minimum and **maximum** are numbers. It is an arrangement for a number of the arrangements between the **minimum** and **maximum** number specified. **minimum** and **maximum** will often be the same, in which case it means choose exactly that many.

For example:

CHO (2 2) (SCD (ENG NWY)) (SCD (ENG DEN)) (SCD (ENG SWE))

Choose two centres from Norway, Denmark and Sweden.

Level 60 : Queries and Insistences

In Level 60, we add the ability to insist on an arrangement, and to ask a question.

press_message = INS (arrangement)

The syntax for **INS** is exactly the same as for **PRP**, and the available replies are also exactly the same. The only difference is that **PRP** should be seen as a suggestion, where as **INS** is much more forceful, and implies that relations between the two powers will be worsened if the **arrangement** is not accepted. If **arrangement** is something that the sending power can enforce on the board (e.g. an order for one of his own units), then the receiving power should expect it to happen whether he likes it or not.

press_message = QRY (arrangement)

The syntax for **QRY** is the same as for **PRP** and **INS**. **QRY** is asking a question – e.g. is there an alliance, rather then proposing – e.g. how about an alliance. **QRY** is responded to in exactly the same way as **PRP** and **INS**. However, the following extra responses are also available for **QRY** :

reply = THK (QRY (arrangement))

I think that **arrangement** is true.

reply = THK (NOT (QRY (arrangement)))

I think that **arrangement** is not true.

reply = FCT (QRY (arrangement))

arrangement is true.

reply = FCT (NOT (QRY (arrangement)))

arrangement is not true.

reply = IDK (QRY (arrangement))

I don't know.

press_message = SUG (arrangement)

The syntax for **SUG** is the same as for **QRY**. **SUG** is suggesting that something is in the mutual interest of the party sending, and all parties receiving the message, but is not something they can directly influence. For instance, Italy may want to say to Austria that it is in their mutual interest for Russia to order to the Black Sea. Replies to **SUG** are the same as for **PRP**. However, the response is an indication of agreement or disagreement, rather than an indication as to whether **arrangement** will actually happen.

THK and **FCT** can also be used as a message in their own right, to pass information without first being queried for the information. **FCT** has already been introduced in level 10.

press_message = THK (arrangement)

I think **arrangement** is true.

~~press_message = FCT (arrangement)~~

~~**arrangement** is true.~~

No reply is expected to ~~either of~~ these.

Note that **THK (QRY (arrangement))** and **THK (arrangement)** have exactly the same meaning, except that the former is the response to a query where as the latter is volunteered without being asked. The same applies to **FCT (QRY (arrangement))** and **FCT (arrangement)**.

Level 70 : Requests for suggestions

In Level 70, we add the ability to ask for suggestions. Requests for suggestions come in two forms :

press_message = WHT (unit)

What should **unit** be ordered to do? This can either be for the sending or a receiving powers unit. In the former case, it is a request for a suggestion. In the latter case, it is a request for information.

WHT messages should be replied to using **PRP**, **INS** to propose or insist on a move (as if it was an initial proposal/insistence, rather than a response to a **WHT**), **IDK (WHT (unit))** if the receiving power doesn't have a suggestion, or with **BWX (WHT (unit))** if the receiving power doesn't want to tell.

press_message = HOW (province)

How do you think we should attack **province**. This should be responded to with **PRP** or **INS** to propose or insist on moves, or with **REJ (HOW (province))** to indicate that the province should not be attacked, **IDK (HOW (province))** if the receiving power doesn't have a suggestion, or with **BWX (HOW (province))** if the receiving power doesn't want to tell.

press_message = HOW (power)

How do you think we should attack **power**. Responses are the same as for **HOW (province)**. **power** must not be an eliminated power.

Level 80 : Accusations

Level 80 adds the ability to accuse another power of going back on its word.

press_message = EXP (turn) (press_message)
press_message = EXP (turn) (reply)

Explain your moves in **turn** given **press_message** that you previously sent.

The standard replies can be used, including :

reply = YES (EXP (turn) (press_message))
reply = YES (EXP (turn) (reply))

Yes I know (no explanation given)

reply = REJ (EXP (turn) (press_message))
reply = REJ (EXP (turn) (reply))

I didn't send **press_message**

reply = IDK (EXP (turn) (press_message))
reply = IDK (EXP (turn) (reply))

There is nothing to explain – they don't conflict

In addition, the following is available.

reply = SRY (EXP (turn) (press_message))
reply = SRY (EXP (turn) (reply))

I'm sorry.

Especially after a **YES** or **BWX** message, it will be common for the next message to be **INS (NOT (PCE (power power ...)))**.

Level 90 : Future discussions

Level 90 adds the ability to talk about the future. To do this, one command is added :

arrangement = FOR (turn) (arrangement)

arrangement = FOR ((start_turn) (end_turn)) (arrangement)

For the **turn** specified, or for all turns from **start_turn** to **end_turn** inclusive, **arrangement** applies. **turn** is of the form **phase year**. The timing specified by **FOR** overrides any timing indicated by **arrangement**.

The **arrangement** parameter of the **FOR** must not include a further **FOR**, unless the arrangement contains a **press_message** (e.g. in **SND**), and the further **FOR** is inside that **press_message**.

Note that **FOR** and **NOT** must be ordered correctly to get the right meaning. **FOR ((start_turn) (end_turn)) (NOT (arrangement))** means that arrangement is not true for any of the turns indicated, while **NOT (FOR ((start_turn) (end_turn)) (arrangement))** means that arrangement is not true for at least one of the turns indicated.

Level 100 : Conditionals

Level 100 adds the ability to make use conditional operators.

press_message = IFF (condition) THN (press_message)
press_message = IFF (condition) THN (press_message) ELS (press_message)

If the **condition** is met then the **THN** part applies, otherwise the **ELS** part applies.

A **condition** takes the same format as a **arrangement.**

IFF is often used in conjunction with **FOR**. i.e. :

IFF (NOT (XDO ((RUS FLT (STP NCS)) BLD))) THN (PRP(FOR (SPR 1902) (XDO ((ENG FLT NWY) SUP (RUS FLT GOB) MTO SWE)))) ELS (INS(FOR (SPR 1902) (XDO ((ENG FLT NWY) SUP (GER FLT DEN) MTO SWE))))

It is recommended that **IFF** statements are always worded so that the receiving power would prefer the **THN** part to the **ELS** part.

Level 110 : Puppets and Favours

Level 110 adds the ability to trade in favours, and to become a puppet of another power.

arrangement = XOY (power) (power)

Power X owes power Y. The powers listed must not be eliminated powers.

The amount owed is indicated by the context. For instance, a client may ask for "You support me to X, and I owe you.". The amount owed is the value of the support. It's for when the client is asking for something now, and offering to return the favour later, not yet knowing how it will be returned.

arrangement = YDO (power) (unit) (unit) …

The control of the listed **unit**s is given to **power**. If the owner of the units is the sending power then it is allowing the named power to decide the orders for those units (which he should do using **PRP (XDO …)**). The orders for the units must still be entered by the owning power. The power must not be an eliminated power.

Level 120 : Forwarding Press

Level 120 adds the ability to forward press from one power to another. This adds three extra commands to the message syntax :

arrangement = SND (power) (power power …) (press_message)
arrangement = SND (power) (power power …) (reply)

This is a request for the first **power** to send the message given. Eliminated powers must not be included in any part of the message.

arrangement = FWD (power power …) (power) (power)

This is either an arrangement for any messages received from any of the powers in the first parameter to the power in the second parameter to be forwarded to the power in the third parameter. Eliminated powers must not be included in any part of the message. Powers should not appear twice in the message. The arrangement is continuous (i.e. it isn't just for the current turn).

arrangement = BCC (power) (power power …) (power)

This is exactly the same as **FWD**, except that the sending and receiving powers are now the first and third parameters. This is an arrangement to forward everything sent, rather than everything received.

arrangement = NOT (FWD (power power …) (power) (power))
arrangement = NOT (BCC (power) (power power …) (power))

This is cancels a previously agreed **FWD or BCC** arrangement.

press_message = FRM (power) (power power …) (press_message)
press_message = FRM (power) (power power …) (reply)

This is a message to inform a power of a message that has been sent or received. No response is required. Eliminated powers must not be included in any part of the message. There is of course no guarantee that the forwarded message is genuine.

Care should be taken to avoid message passing loops. In general, a client should not forward a message that already contains the segment **(FRM (you) (power you are forwarding to) (press_message))**.

Level 130 : Explanations

Level 130 adds the ability to ask why a power thinks something. In response to a **THK** or **FCT** message, a client can reply with :

reply = WHY (THK (arrangement))

reply = WHY (FCT (arrangement))

This is a request for an explanation as to the reason why sender thinks the **arrangement** in the **THK** or **FCT** message is true. Available replies are **BWX** if the client don't want to tell, **REJ** if the client wants to deny knowledge of the **THK** or **FCT** message, or another **THK** or **FCT** message to give further information. Additionally, a client can also give any other message which gives further information (most notably a **FRM** message received from another power). Finally, there is one new response available :

reply = POB (WHY (…))

The position on the board, or the previous moves, suggests/implies it.

WHY can also be used in other contexts:

reply = WHY (SUG (arrangement))

Why do you think that would be good for us?

reply = WHY (PRP (arrangement))
reply = WHY (INS (arrangement))

Why are you proposing that?

Level 130 also adds some additional uses of **IDK** :

reply = IDK (PRP (arrangement))
reply = IDK (INS (arrangement))

I don't know whether I'll accept that. Throw in something else to make it worthwhile.

reply = IDK (SUG (arrangement))

I don't really know whether I desire that.

Level 8000 : Free Text Press

This level allows natural language press. Both **message** and **reply** can be any string of ASCII characters.

Appendix A: Complete List of Tokens

The following is a complete alphabetical list of tokens specified in the Diplomacy AI Development Environment Syntax.

Token	Token Type	Usage	Meaning
ADR	Province	Sea	Adriatic Sea
AEG	Province	Sea	Aegean Sea
ALB	Province	Coastal	Albania
ALY	Press		Ally
AMY	Unit Type		Army
AND	Press		Logical AND
ANK	Province	Coastal Supply Centre	Ankara
AOA	Parameter	HLO	Any Orders Allowed
APU	Province	Coastal	Apulia
ARM	Province	Coastal	Armenia
AUS	Power		Austria
AUT	Phase		Fall Retreats
BAL	Province	Sea	Baltic Sea
BAR	Province	Sea	Barents Sea
BCC	Press		Request to Blind Carbon Copy
BEL	Province	Coastal Supply Centre	Belgium
BER	Province	Coastal Supply Centre	Berlin
BLA	Province	Sea	Black Sea
BLD	Order	Build Phase	Build
BNC	Order Note	ORD	Move Bounced
BOH	Province	Inland	Bohemia
BPR	Order Note	THX	REMOVED
BRE	Province	Coastal Supply Centre	Brest
BTL	Parameter	HLO	Build Time Limit
BUD	Province	Inland Supply Centre	Budapest
BUL	Province	Bicoastal Supply Centre	Bulgaria
BUR	Province	Inland	Burgundy
BWX	Press		None of Your Business
CCD	Command	Server to Client	Power in Civil Disorder
CCL	Press		Cancel
CHO	Press		Choose
CLY	Province	Coastal	Clyde
CON	Province	Coastal Supply Centre	Constantinople
CST	Order Note	THX	No Coast Specified
CTO	Order	Movement Phase	Move by Convoy to
CUT	Order Note	ORD	Support Cut
CVY	Order	Movement Phase	Convoy
DEN	Province	Coastal Supply Centre	Denmark
DMZ	Press		Demiliterised Zone
DRW	Command / Press	Client <---> Server	Draw
DSB	Order	Retreat Phase	Disband
DSR	Order Note	ORD	Convoy Disrupted
EAS	Province	Sea	Eastern Mediterranean Sea
ECH	Province	Sea	English Channel
ECS	Coast		East Coast
EDI	Province	Coastal Supply Centre	Edinburgh
ELS	Press	IFF	Else
ENG	Power		England
ERR	Parameter	HUH	Error location
ESC	Order Note	THX	Not an Empty Supply Centre
EXP	Press		Explain
FAL	Phase		Fall Movements
FAR	Order Note	THX	Not Adjacent
FCT	Press		Fact
FIN	Province	Coastal	Finland

FLD	Order Note	ORD	REMOVED
FLT	Unit Type		Fleet
FOR	Press		For specified Turn
FRA	Power		France
FRM	Command / Press	Server to Client	Message From
FWD	Press		Request to Forward
GAL	Province	Inland	Galecia
GAS	Province	Coastal	Gascony
GER	Power		Germany
GOB	Province	Sea	Gulf of Bothnia
GOF	Command	Client to Server	Go Flag
GOL	Province	Sea	Gulf of Lyons
GRE	Province	Coastal Supply Centre	Greece
HEL	Province	Sea	Helgoland Bight
HLD	Order	Movement Phase	Hold
HLO	Command	Server to Client	Hello (Start of Game)
HOL	Province	Coastal Supply Centre	Holland
HOW	Press		How to attack
HSC	Order Note	THX	Not a Home Supply Centre
HST	Command	Client to Server	History
HUH	Command / Press	Server to Client	Syntax Error / Not Understood
IAM	Command	Client to Server	I am
IDK	Press		I Don't Know
IFF	Press		If
INS	Press		Insist
ION	Province	Sea	Ionian Sea
IRI	Province	Sea	Irish Sea
ITA	Power		Italy
KIE	Province	Coastal Supply Centre	Kiel
LOD	Command	Server to Client	Load Game
LON	Province	Coastal Supply Centre	London
LVL	Parameter	HLO	Level (Language Level)
LVN	Province	Coastal	Livonia
LVP	Province	Coastal Supply Centre	Liverpool
MAO	Province	Sea	Mid Atlantic Ocean
MAP	Command	Server to Client	Map to be used for this game
MAR	Province	Coastal Supply Centre	Marseilles
MBV	Order Note	THX	Might Be Valid
MDF	Command	Client <---> Server	Map definition
MIS	Command	Server to Client	Missing Orders
MOS	Province	Inland Supply Centre	Moscow
MRT	Parameter	NOW	Must Retreat to
MTL	Parameter	HLO	Movement Time Limit
MTO	Order	Movement Phase	Move To
MUN	Province	Inland Supply Centre	Munich
NAF	Province	Coastal	North Africa
NAO	Province	Sea	North Atlantic Ocean
NAP	Province	Coastal Supply Centre	Naples
NAR	Press		No Agreement
NAS	Order Note	THX	Not At Sea
NCS	Coast		North Coast
NMB	Order Note	THX	No More Builds Allowed
NME	Command	Client to Server	Name
NMR	Order Note	THX	No More Retreats Allowed
NOT	Command / Press	Client <---> Server	Logical NOT
NOW	Command	Client <---> Server	Current Position
NPB	Parameter	HLO	No Press During Builds
NPR	Parameter	HLO	No Press During Retreats
NRN	Order Note	THX	No Retreat Needed

NRS	Order Note	THX	Not the Right Season
NSC	Order Note	THX	Not a Supply Centre
NSF	Order Note	THX	No Such Fleet
NSO	Order Note	ORD	No Such Order
NSP	Order Note	THX	No Such Province
NSU	Order Note	THX	No Such Unit
NTH	Province	Sea	North Sea
NVR	Order Note	THX	Not a Valid Retreat
NWG	Province	Sea	Norwegian Sea
NWY	Province	Coastal Supply Centre	Norway
NYU	Order Note	THX	Not Your Unit
OBS	Command	Client to Server	Observer
OCC	Press		Occupy
OFF	Command	Server to Client	Turn Off (Exit)
ORD	Command	Server to Client	Order Results
ORR	Press		Logical OR
OUT	Command	Server to Client	Power is Eliminated
PAR	Province	Inland Supply Centre	Paris
PCE	Press		Peace
PDA	Parameter	HLO	Partial Draws Allowed
PIC	Province	Coastal	Picardy
PIE	Province	Coastal	Piedmont
POB	Press		Position on Board
POR	Province	Coastal Supply Centre	Portugal
PRN	Command	Server to Client	Parenthesis error
PRP	Press		Propose
PRU	Province	Coastal	Prussia
PTL	Parameter	HLO	Press Time Limit
QRY	Press		Query
REJ	Command / Press	Server to Client	Reject
REM	Order	Build Phase	Remove
RET	Order Note	ORD	Unit must retreat
ROM	Province	Coastal Supply Centre	Rome
RTL	Parameter	HLO	Retreat Time Limit
RTO	Order	Retreat Phase	Retreat to
RUH	Province	Inland	Ruhr
RUM	Province	Coastal Supply Centre	Rumania
RUS	Power		Russia
SCD	Press		Supply Centre Distribution
SCO	Command	Client <---> Server	Supply Centre Ownership
SCS	Coast		South Coast
SER	Province	Inland Supply Centre	Serbia
SEV	Province	Coastal Supply Centre	Sevastopol
SIL	Province	Inland	Silesia
SKA	Province	Sea	Skaggerack
SLO	Command	Server to Client	Solo
SMR	Command	Server to Client	Summary
SMY	Province	Coastal Supply Centre	Smyrna
SND	Command / Press	Client to Server	Send Message
SPA	Province	Bicoastal Supply Centre	Spain
SPR	Phase		Spring Movement
SRY	Press		Sorry
STP	Province	Bicoastal Supply Centre	St Petersburg
SUB	Command	Client to Server	Submit Order
SUC	Order Note	ORD	Order Succeeds
SUG	Press		Suggest
SUM	Phase		Spring Retreats
SUP	Order	Movement Phase	Support
SVE	Command	Server to Client	Save Game

SWE	Province	Coastal Supply Centre	Sweden
SYR	Province	Coastal	Syria
THK	Press		Think
THN	Press	IFF	Then
THX	Command	Server to Client	Thanks for the order
TME	Command	Client <---> Server	Time to Deadline
TRI	Province	Coastal Supply Centre	Trieste
TRY	Press		Try the following tokens
TUN	Province	Coastal Supply Centre	Tunis
TUR	Power		Turkey
TUS	Province	Coastal	Tuscany
TYR	Province	Inland	Tyrolia
TYS	Province	Sea	Tyrrhenian Sea
UKR	Province	Inland	Ukraine
UNO	Parameter	SCO	Unowned
UNT	Press	OCC	Unit
VEN	Province	Coastal Supply Centre	Venice
VIA	Order	Movement Phase	Move via
VIE	Province	Inland Supply Centre	Vienna
VSS	Press	ALY	Versus
WAL	Province	Coastal	Wales
WAR	Province	Inland Supply Centre	Warsaw
WES	Province	Sea	Western Mediterranean Sea
WHT	Press		What to do with
WHY	Press		Why
WIN	Phase		Fall Builds
WVE	Order	Build Phase	Waive
XDO	Press		Moves to do
XOY	Press		X owes Y
YDO	Press		You provide the order for these units
YES	Command / Press	Server to Client	Accept
YOR	Province	Coastal	Yorkshire
YSC	Order Note	THX	Not Your Supply Centre
integer	Parameter	Various	Used for years, number of builds, etc.

Appendix B: Language definition.

This appendix lists the entire language in a more machine-oriented form, similar to BNF. The first two types, client_server_message and server_client_message are the complete messages that can be sent. They are then composed of tokens and other types.

To read the table, the category in the first column can be replaced by any of the combinations in the third column. The language level in the second column indicates the minimum language level at which this substitution is allowed. OBS indicates any level including an Observer.

Type	Level	Composition
client_server_message	OBS	NME (string) (string)
	OBS	OBS
	OBS	IAM (power) (number)
	OBS	MAP
	OBS	MDF
	OBS	YES (acknowledgable_command)
	OBS	REJ (acknowledgable_command)
	OBS	NOW
	OBS	SCO
	OBS	HST (turn)
	OBS	TME (number)
	OBS	TME
	OBS	ADM (string) (string)
	OBS	PRN (bad_bracketed_sequence)
	OBS	HUH (any_token_sequence)

	0	HLO	
	0	SUB (order) (order) ...	
	0	SUB (turn) (order) (order) ...	
	0	NOT (negatable_command)	
	0	MIS	
	0	GOF	
	0	ORD	
	0	DRW	
	10	DRW (power power_list)	
	10	send_message	
server_client_message	OBS	YES (client_command)	
	OBS	YES (client_request)	
	OBS	REJ (client_request)	
	OBS	REJ (rejectable_client_command)	
	OBS	MAP (string)	
	OBS	MDF (power_list) (mdf_provinces) (mdf_adjacency_list)	
	OBS	HLO (power) (number) (variant)	
	OBS	NOW (turn) (unit_with_location_and_mrt) (unit_with_location_and_mrt) ...	
	OBS	SCO (sco_entry) (sco_entry) ...	
	0	THX (order) (order_note)	
	0	MIS (unit_with_location_and_mrt) (unit_with_location_and_mrt) ...	
	0	MIS (number)	
	OBS	ORD (turn) (order) (compound_order_result)	
	0	SVE (string)	
	0	LOD (string)	
	OBS	OFF	
	OBS	TME (number)	
	OBS	PRN (bad_bracketed_sequence)	
	OBS	HUH (any_token_sequence)	
	OBS	CCD (power)	
	OBS	NOT (negated_server_message)	
	OBS	ADM (string) (string)	
	OBS	SLO (power)	
	OBS	DRW	
	OBS	SMR (turn) (power_summary) (power_summary) ...	
	10	DRW (power power_list)	
	10	OUT (power)	
	10	FRM (power number) (power_list) (press_message)	
	10	FRM (power number) (power_list) (reply)	
acknowedgable_command	OBS	MAP (string)	
	OBS	SVE (string)	
any_token_sequence	OBS	Any sequence of tokens where the brackets all match correctly	
bad_bracketed_sequence	OBS	A sequence of tokens where the bracketing is incorrect	
coast	OBS	Any token from the Coast category	
compound_order_result	OBS	order_note	
	OBS	order_result	
	OBS	order_note RET	
	OBS	order_result RET	
client_command	OBS	OBS	
	OBS	NOT (TME)	
client_request	OBS	NME (string) (string)	
	OBS	IAM (power) (number)	
	0	NOT (GOF)	
	0	GOF	
	OBS	TME (number)	
	OBS	DRW	
	OBS	NOT (negated_client_request)	
	10	DRW (power power_list)	
	10	send_message	
explanation	80	EXP (turn) (reply)	
future_offer	10	PCE (power power_list)	

	10	ALY (power_list) VSS (power_list)
	10	DRW
	10	SLO (power)
	10	NOT (future_offer)
	20	XDO (order)
	20	DMZ (power_list) (province_list)
	40	SCD (sc_ownership_list) (sc_ownership_list) ...
	40	OCC (occ_unit_with_location) (occ_unit_with_location) ...
	50	AND (future_offer) (future_offer) ...
	50	ORR (future_offer) (future_offer) ...
	50	CHO (number number) (future_offer) (future_offer) ...
	110	XOY (power) (power)
	110	YDO (power) (unit_with_location) (unit_with_location) ...
	120	SND (power) (power_list) (press_message)
	120	SND (power) (power_list) (reply)
	120	FWD (power_list) (power) (power)
	120	BCC (power) (power_list) (power)
logical_operator	30	AND (arrangement) (arrangement) ...
	30	ORR (arrangement) (arrangement) ...
mdf_adjacencies	OBS	(mdf_province_adjacencies) (mdf_province_adjacencies) ...
mdf_centre_list	OBS	power province_list
	OBS	UNO province_list
	OBS	(power_list) province_list
mdf_coast	OBS	FLT coast
mdf_coast_adjacencies	OBS	unit_type mdf_province mdf_province ...
	OBS	(mdf_coast) mdf_province mdf_province ...
mdf_province	OBS	province
	OBS	(province_and_coast)
mdf_province_adjacencies	OBS	province (mdf_coast_adjacencies) (mdf_coast_adjacencies) ...
mdf_province_list	OBS	mdf_province mdf_province ...
mdf_provinces	OBS	(mdf_supply_centres) (province_list)
mdf_supply_centres	OBS	(mdf_centre_list) (mdf_centre_list) ...
negatable_command	OBS	SUB (order)
	OBS	SUB
	OBS	GOF
	OBS	DRW
	OBS	DRW (power power_list)
	OBS	TME
	OBS	TME (number)
negatable_query	60	QRY (arrangement)
	60	NOT (query)
negated_client_request	OBS	TME (number)
	OBS	DRW
	10	DRW (power power_list)
negated_server _message	OBS	CCD (power)
	OBS	TME (number)
number	OBS	Any token from the Number categories
arrangement	10	PCE (power power_list)
	10	ALY (power_list) VSS (power_list)
	10	DRW
	10	SLO (power)
	10	NOT (arrangement)
	10	NAR (arrangement)
	20	XDO (order)
	20	DMZ (power_list) (province_list)
	40	SCD (sc_ownership_list) (sc_ownership_list) ...
	40	OCC (occ_unit_with_location) (occ_unit_with_location) ...
	50	AND (arrangement) (arrangement) ...
	50	ORR (arrangement) (arrangement) ...
	50	CHO (number number) (arrangement) (arrangement) ...
	90	FOR (turn) (future_offer)

		90	FOR (period) (future_offer)
		110	XOY (power) (power)
		110	YDO (power) (unit_with_location) (unit_with_location) ...
		120	SND (power) (power_list) (press_message)
		120	SND (power) (power_list) (reply)
		120	FWD (power_list) (power) (power)
		120	BCC (power) (power_list) (power)
order		OBS	(unit_with_location) HLD
		OBS	(unit_with_location) MTO mdf_province
		OBS	(unit_with_location) SUP (unit_with_location)
		OBS	(unit_with_location) SUP (unit_with_location) MTO province
		OBS	(unit_with_location) CVY (unit_with_location) CTO province
		OBS	(unit_with_location) CTO province VIA (province_list)
		OBS	(unit_with_location) RTO mdf_province
		OBS	(unit_with_location) DSB
		OBS	(unit_with_location) BLD
		OBS	(unit_with_location) REM
		OBS	power WVE
order_note		OBS	MBV
		OBS	FAR
		OBS	NSP
		OBS	NSU
		OBS	NAS
		OBS	NSF
		OBS	NSA
		OBS	NYU
		OBS	NRN
		OBS	NVR
		OBS	YSC
		OBS	ESC
		OBS	HSC
		OBS	NSC
		OBS	CST
		OBS	NMB
		OBS	NMR
		OBS	NRS
order_result		OBS	SUC
		OBS	BNC
		OBS	CUT
		OBS	DSR
		OBS	NSO
period		90	(turn) (turn)
power		OBS	Any token from the Power category
power_list		OBS	power power ...
power_summary		OBS	power (string) (string) number
		OBS	power (string) (string) 0 number
press_message		10	PRP (arrangement)
		10	TRY (try_parameters)
		10	CCL (press_message)
		30	PRP (logical_operator)
		60	INS (arrangement)
		60	QRY (arrangement)
		60	SUG (arrangement)
		60	THK (arrangement)
		610	FCT (arrangement)
		70	WHT (unit_with_location)
		70	HOW (province)
		70	HOW (power)
		80	EXP (turn) (reply)
		100	IFF (arrangement) THN (press_message)
		100	IFF (arrangement) THN (press_message) ELS (press_message)

	120	FRM (power number) (power_list) (press_message)
	120	FRM (power number) (power_list) (reply)
	8000	string
province	OBS	Any token from any of the Province categories
province_list	OBS	province province ...
province_and_coast	OBS	province coast
query	60	QRY (arrangement)
reply	10	YES (press_message)
	10	REJ (press_message)
	10	BWX (press_message)
	10	HUH (any_token_sequence)
	60	THK (negatable_query)
	60	FCT (negatable_query)
	60	IDK (query)
	80	YES (explanation)
	80	REJ (explanation)
	80	IDK (explanation)
	80	SRY (explanation)
	130	WHY (think_and_fact)
	130	POB (why_sequence)
	130	WHY (SUG (arrangement))
	130	WHY (PRP (arrangement))
	130	WHY (INS (arrangement))
	130	IDK (PRP (arrangement))
	130	IDK (INS (arrangement))
	130	IDK (SUG (arrangement))
	10	press_message
rejectable_client_command	OBS	HLO
	OBS	NOW
	OBS	SCO
	OBS	HST (turn)
	0	SUB (order) (order) ...
	OBS	ORD
	OBS	TME
	OBS	ADM (string) (string)
sc_ownership_list	OBS	power province_list
	OBS	UNO province_list
sco_entry	OBS	power province_list
season	OBS	Any token from the Season category
send_message	10	SND (power_list) (press_message)
	10	SND (power_list) (reply)
	10	SND (turn) (power_list) (press_message)
	10	SND (turn) (power_list) (reply)
string	OBS	Sequence of tokens from the ASCII category
think_and_fact	130	THK (arrangement)
	130	FCT (arrangement)
try_parameters	10	
	10	try_token try_token ...
try_token (client to server)	10	PRP
	10	PCE
	10	ALY
	10	VSS
	10	DRW
	10	SLO
	10	NOT
	10	NAR
	10	YES
	10	REJ
	10	BWX
	10	CCL
	10	XDO

	10	DMZ
	10	AND
	10	ORR
	10	SCD
	10	OCC
	10	INS
	10	QRY
	10	THK
	10	FCT
	10	IDK
	10	SUG
	10	WHT
	10	HOW
	10	EXP
	10	SRY
	10	FOR
	10	IFF
	10	THN
	10	ELS
	10	XOY
	10	YDO
	10	FRM
	10	FWD
	10	SND
	10	WHY
	10	POB
try_token (server to client)	10	PRP
	10	PCE
	10	ALY
	10	VSS
	10	DRW
	10	SLO
	10	NOT
	10	NAR
	10	YES
	10	REJ
	10	BWX
	10	CCL
	20	XDO
	20	DMZ
	30	AND
	30	ORR
	40	SCD
	40	OCC
	60	INS
	60	QRY
	60	THK
	610	FCT
	60	IDK
	60	SUG
	70	WHT
	70	HOW
	80	EXP
	80	SRY
	90	FOR
	100	IFF
	100	THN
	100	ELS
	110	XOY
	110	YDO
	120	FRM

	120	FWD
	120	SND
	130	WHY
	130	POB
turn	OBS	season number
unit_type	OBS	Any token from the Unit category
unit_with_location	OBS	power unit_type province
	OBS	power unit_type (province_and_coast)
occ_unit_with_location	50	power unit_type province
	50	power unit_type (province_and_coast)
	50	power UNT province
unit_with_location_and_mrt	OBS	power unit_type province
	OBS	power unit_type (province_and_coast)
	OBS	power unit_type province MRT (mdf_province_list)
	OBS	power unit_type (province_and_coast) MRT (mdf_province_list)
variant	OBS	(variant_option) (variant_option) ...
variant_option	OBS	LVL number
	OBS	MTL number
	OBS	RTL number
	OBS	BTL number
	OBS	AOA
	OBS	DSD
	10	PDA
	10	PTL number
	10	NPR
	10	NPB
why_sequence	130	WHY (think_and_fact)