WitpStaff

(C) 2006-2009 M.Baumeister

Table of Contents

1 Requirements....2
2 Configuration....2
3 Running WitpStaff....3
3.1 Initialization....3
3.2 Reading in a save file....4
3.3 Ending the program....4
3.4 Creating another database....5
3.5 If anything goes wrong during initialization....5
3.6 PBEM security....5
4 Displayed information....6
4.1 Storage Tab....6
4.1.a The History Graph....7
4.2 Industry Tab....8
4.2.a ClusterBases.csv....9
4.2.b Spoilage....9
4.3 Aircraft classes....9
4.3.a The performance values....11
4.4 LCU targets Tab....12
4.5 Ship Updates Tab....12
4.6 Ship Classes Tab....13
4.7 Ship History Tab....14
4.8 Leaders Tab....15
4.8.a Preferred queries and leader-prefs.csv....16
4.9 Map Overviews Tab....16
4.10 Events Tab....16
4.11 Inconsistencies Tab....17
5 Map Overviews....18
5.1 Naval transport....18
5.1.a Side column....18
5.2 Naval search....19
5.3 Support....20
5.4 Supply....21
5.5 Attack value....21
5.6 Other Plugins....22
6 Warnings....22
7 Known Bugs....22
8 Extensions....22
9 Copyright & Co....23
10 Thanks....23

This program started due to my dissatisfaction with the GUI of "War in the Pacific". The game features a multitude of decision possibilities for the players, figuratively hundreds of knobs to turn but failed to provide basic information necessary to make informed decisions without reverting to keeping records on paper or spreadsheet.

Thus this program analyses parts of the save file of the game, retrieves any useful information into a SQL database (Thanks HSQLDB folks!) and then tries to display it in a clean and informative way to the user (Thanks partially to the SWT people). It thereby partially replaces the staff a typical WW II chief of staff would have. This should help especially players of the Japanese side which are tasked with non-trivial logistics by the game without providing much usefully processed information to them.

1 Requirements

WitpStaff is installed automatically in the tools/witpstaff subdirectory of WitpAE. It contains all the necessary libraries and configuration files (see below on how to fiddle with them). Nevertheless it needs one thing which is not included:

- Java 1.5 or better (download the JRE from http://java.sun.com/javase/downloads/index.jsp if it is not already installed in your system anyway)

2 Configuration

WitpStaff puts most of its configuration into .csv files (Comma Separated Values) as the used database engine can easily read them. While you don't *have to* change most of them, they allow you to adapt WitpStaff more to your preferences. They can be edited using any text editor (e.g. notepad) but then you run the risk of introducing syntax errors (e.g. forgetting commas or apostrophes). I therefore recommend editing with a spreadsheet program such as Excel or OpenOffice Calc. Make sure to save them again as .csv files and not in the native format of the spreadsheet program.

The .csv files are copied from WitpStaff's main directory into the directory of each database upon database initialization. Thus if you want to change them for all future databases, you should change them in WitpStaff's main directory. If you want to change them only for one database (i.e. one game), you should change them in the subdirectory of that database.

Let's go through the individual files in the following:

1. To make full use of the Industry tab (see chapter 4.2) you need a “clusters.csv” and a “clusterbases.csv” file *for your specific scenario* and put them in the newly created directory. The files provided in the main directory of WitpStaff are for the main scenario (scenario 1). If some scenario developers or other users have provided additional files, they will be in the “clusterDefinitions” subdirectory. To use them, copy them into the main directory of WitpStaff *before* initializing the database (see chapter 3). It might be sufficient to copy them into the subdirectory created for your database after initialization, but I give no guarantee.
   If you are using the wrong 'cluster' files for a scenario, potentially false information will be displayed in the Industry tab. If no appropriate files are available, you might want to either delete them before database initialization or edit them to correct the base-cluster assignments. More information on the format of the file can be found in the section on the Industry tab (chapter 4.2).
2. The Leader-Tab (see chapter 4.8) provides a number of presets to speed up finding a leader with attributes weighted to your liking. These presets are read at each start of WitpStaff from the leader-prefs.csv file. If you want to change them or add new ones

Not required but useful can be adaptations of the leader-prefs.csv and the acfactors.csv files to your liking. See the chapter on the leader tab and on the aircraft classes tab. The ships4history.csv file defines which ship classes will be stored in the database for the Ship History Tab.
If something goes wrong during initialization and you want to start over, you have to remove the

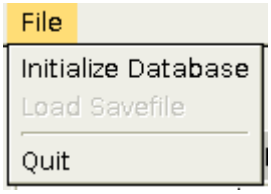
database directory and initialize a new database. If you changes .csv files in that database directory, it might be a good idea to save them before deleting the directory.

3 Running WitpStaff

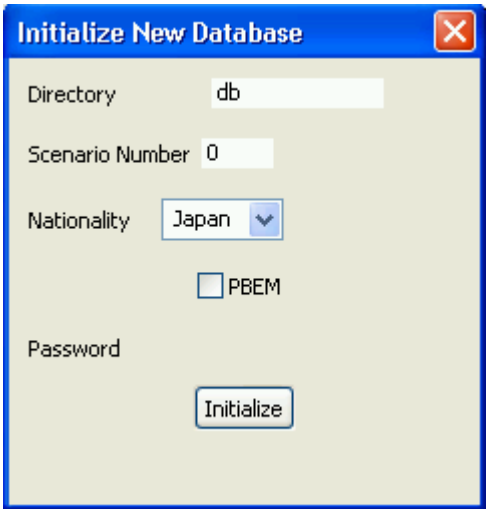
3.1 Initialization

Click on the witpStaff.bat file (or start it from a CMD.EXE or similar console). Directly clicking on the witpDecoder2.jar file will also work but is unsupported as you won't directly see some error output from WitpStaff.

After starting the program for the first time, the database is uninitialized. This results in the menu not allowing you to load a save game:



Clicking on Initialize Database pops up a window into which you have to add data as explained below.



- The directory you want the database to reside in. The directory will be relative to witpStaff's directory and must not yet exist.
- The number of the scenario which was used to start the game you want to load into witpStaff. This scenario has to be available in your witp scen directory.
- Whether you are playing Japan or the Allies. If you want to analyse both sides in a game against the AI you will have to create two databases and put those into two directories. If you are playing a PBEM you *have to* select your side here as you are lacking the password for the other side.
- Whether the save games are from a PBEM or not. If you check this mark a further field will occur into which you must enter the password you use in WitPAE for that PBEM. Note that this password will be stored in WitpStaff's database and anyone a little knowledgeable in SQL can read it from there. Therefore do not send WitpStaff's database (anything in the directory you specified above) to your opponent.

When finished, click Initialize. This takes some time.

After the program has initialized itself, it will say so, shut down and ask you to start it again[1]. Please give the program a few seconds for dumping the database before restarting. Otherwise you might get an error telling you the database is not accessible.

3.2 Reading in a save file

After a database was initialized it is possible to read save files into it. Select 'File → **Load Savefile**' from the menu and then select the correct file to load in the file selector window that pops up. Be sure to read in the *correct* file as you can never go back 'before' the file you read (i.e. you cannot read in turns before a turn already read in).

Reading in a save file takes some time (from ca. 5 to 30 seconds depending on computer) and there is currently sparse feedback (a progress bar is missing). So don't despair, it is unlikely that the tool is hanging in an endless loop.

Yet, reading in a save game file is the most probable reason for encountering an error. You can either get known errors due to some inconsistency in the save game file (that's Matrixgames' fault) or exceptions caused by me misunderstanding the save game format. The former will show up as "Errors" or "Warnings" which most of the time can be cancelled to never occur again until you restart WitpStaff. The latter show up as "Exceptions". Complain about the former to Matrixgames or your scenario author (and try to fix them in-game) and complain about the latter to me by sending/posting the stack backtrace you find in the witpdecoder.log[2] file. Be aware that sometimes bugs within WitpStaff masquerade as save game problems so you might want to check in-game if an inconsistency WitpStaff complains about actually exists.

You can read in several save game files in one go (with reduced pop-up windows) by using the "File->**Load Savefile batch**" menu item. You will then be presented with a file selector to select the *first* file of the batch (not all of them). For batch loading to work all file names must follow the following convention: SomeText0123MoreText. This means:

- The save files names starts with a character (and preferably also ends with one since ending with the number is in principle possible but untested).
- The text parts of the save files names have to be the same for all files of the batch
- There is a number within each save file name. This number has leading zeros (i.e. if you ever want that number to go into the hundreds you better now use "012" for twelve and not "12").
- Beginning with the start file all files with consecutive numbers will be read. Loading stops when a save file name with the next number does not exist or when an error occurs.

Basically if you design your save file names similar to the 'official' witpXXX.pws name, it will work. Note that loading will take quite some time. You might see the turn number and the save file name updated in WitpStaff's window. If not (giving back the CPU to SWT just for an update doesn't seem to work consistently) you will simply have to wait. You will also need up to 200MB free space on your hard disk since in this mode WitpStaff will only start to optimize its database if it grows bigger than that.

3.3 Ending the program

An important thing to remember is to end the program using the "**Quit"** entry in the File menu. Do not use the "Window close" X-button provided by MS Windows in the top right corner of each

1 This is because the original witpStaff (known as witpDecoder then) could handle only one database. To ensure everything works fine after initialization, witpStaff requires a restart so it will use the databases in the respective directory from its start.

2 Not the "witp.log" file. That is part of the database..

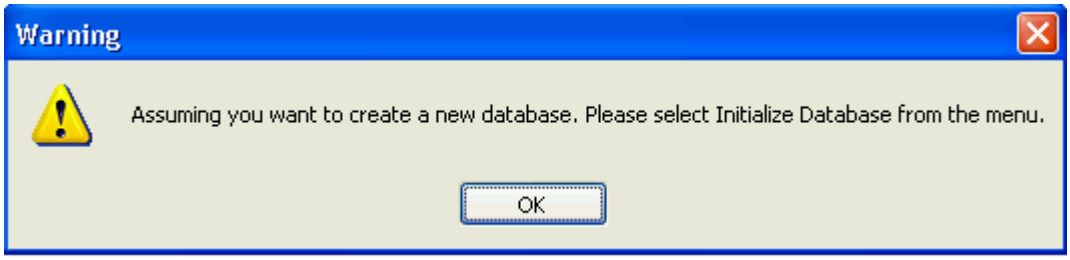
window. If you continuously use that button, the database will never be cleaned up and program start-up will become slower and slower over time.

(In principle it would be of course possible to catch the “Window close” button and then clean up the database. I am only wary that if database cleanup takes too long, Windows might present the user with a “Program not responding” dialog and allow the user to kill WitpStaff. I have no idea what happens if an hsqldb program is killed while cleaning its database. So I don't want people to try out.)

3.4 Creating another database

If you want to use WitpStaff for more than one ongoing game, you need more than one database because each database should only be fed with save game files from one game.

To create another database you must first bring WitpStaff into its 'uninitialized' state as it will otherwise not allow you to initialize another database. This works by selecting the 'File-> **Change DB Directory**' menu entry and then clicking **Cancel** in the directory selection window that pops up. WitpStaff will display



and then allow you to create another database. Elegant, isn't it?

3.5 If anything goes wrong during initialization

You will find that error reports appear as pop-up windows and/or are dumped into the “witpDecoder.log” file. If anything seems to go wrong without any error message, look in that file. Remember that witpDecoder.log is deleted at the beginning of each WitpStaff execution. So copy it somewhere safe if you want to avoid loosing an error message when trying again.

If you installed the tool into the wrong directory, specified a non-existing scenario number, or anything else went wrong, you will have to delete the database directory before trying again. Otherwise WitpStaff will complain about a partially initialized database and not do anything.

If the .csv files themselves turn out to be problematic (like e.g. some CHS files did in WitP time, which contained a bug in the ship classes database featuring a ship with more then 200 torpedoes), you can edit the .csv files copied into the WitpStaff directory and use the edited files for database initialization. For this *remove witpload.exe from the scenario directory*. WitpStaff will fail executing witpload (pop up window informs you about the problem). As fall-back it then uses any WITP*.csv files already in its local directory, so you copy the changed .csv into the new database directory and let WitpStaff continue.

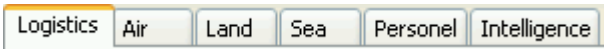
3.6 PBEM security

Several people have asked what WitpStaff means for PBEM security. Basically nothing changes. WitpStaff reads the data from the save game file into its own database and never writes data back. So no change to save game files will be done by WitpStaff. You also need the correct password to retrieve the data from the save game into the database. So no spying is possible either. WitpStaff's database itself is *not* protected though and can be accessed without any password. Thus you might want to avoid that your PBEM opponent gets ahold of any of the witp.* files that WitpStaff generates.

Of course enterprising players could try to hack WitpStaff. But hacking the WitP.exe program has been possible all the time and it seemingly didn't happen. Also I took some measures to make hacking WitpStaff more difficult.

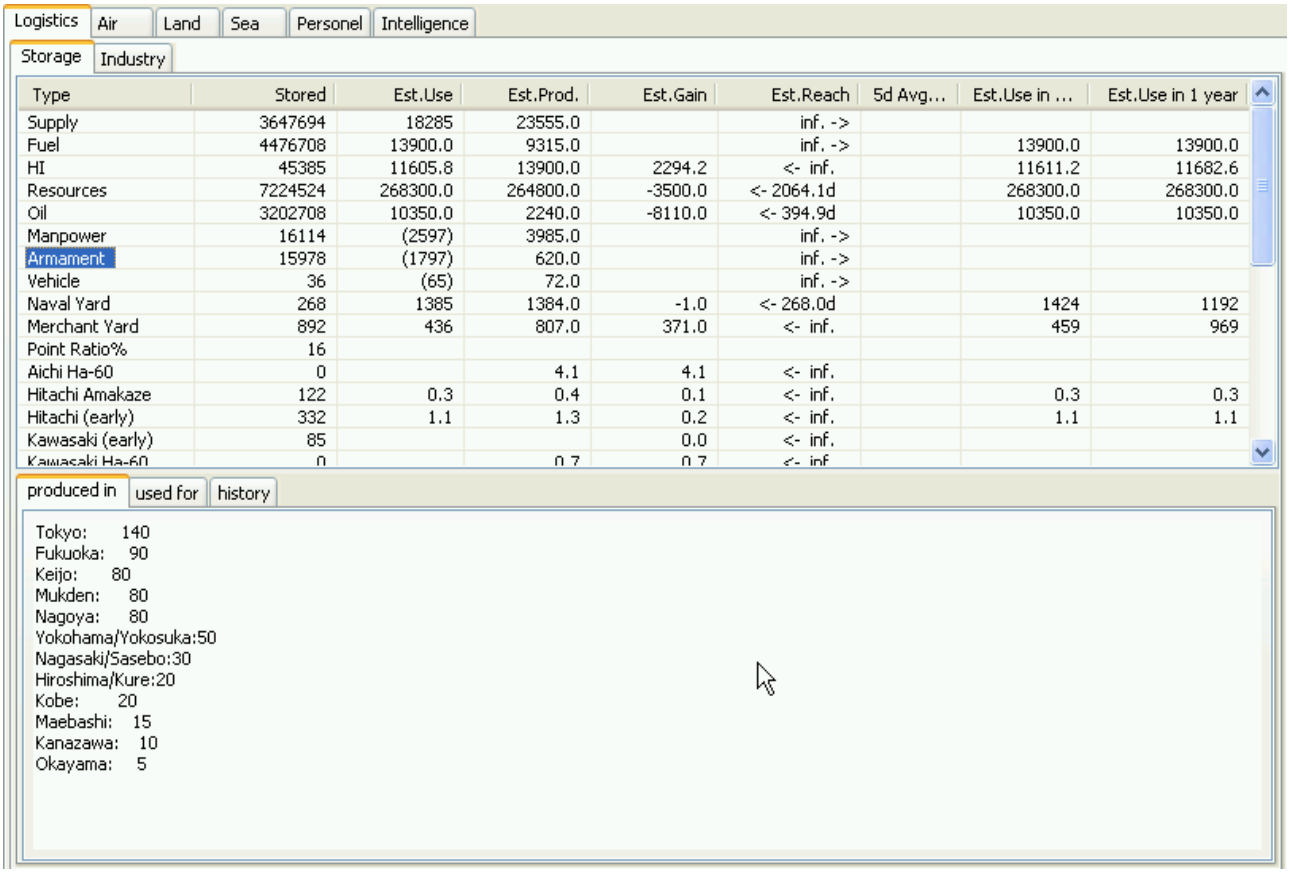
4 Displayed information

The tool displays its information in several tabs in its main window. The tabs are organized hierarchically with the first level representing the different 'officers' your staff has. One officer ... uhm ... tab per general topic will give you information:



Below each of these 'officer' tabs one or more data displays are arranged per topic. If only one tab belongs to that topic, the respective display will be shown immediately, otherwise there will be a tab hierarchy. Now lets look at the tab in detail.

4.1 Storage Tab



This tabs tells you the overall amount of the various **'strategic resources'** (Supply, Fuel, HI, Resources, Engines, ...) you have left, their estimated production and usage as well as the measured change in stock averaged over the last 5 days. It also tries to calculate how long your current stock will reach assuming that the current usage and production continues. For long term planning also an estimate of the usage in one month and in one year[3] is made. Note that this estimate will nearly always be wrong when expanding factories since it assumes that everything will be repaired one step each day which in general is not true (e.g. due to supply shortages, slow repair while researching, ...).

3 This was added in WitpStaff to give a better feeling for the required amount of Naval and Merchant build points. Otherwise players could be tempted to stop yards or even convert them to the other type although some reserve buildup is necessary for a later time.

The “stored” column takes into account also resources *transported* on a ship (in contrast to the game which only counts things at land). Thus data shown here for Supply, Fuel, Oil and Resources should be higher than in game and the values should jump less around when loading and unloading large convoys.

The “Estimated Use” column is calculated in different ways: For some resources it is calculated based on the (hopefully) correct relationships from WitP's economic explanations. It the shows the amount which will probably be used during the *next* turn resolution. For other resources, it impossible to (easily) calculate this and thus WitpStaff estimates the use from what was used during the last turn resolution. These values will be shown in braces. (Note the difference between *next* and *last* turn resolution).

The arrows in the “Est.Reach” column point to the column which was used to calculate the specific reach. An arrow to the left means the reach was calculated using the estimated usage. An arrow to the right means the reach was calculated using the measured gains over the last 5 days. The latter has to be used sometimes since for many products the amount of usage is unknown or very difficult to calculate. But using the measured usage may lead to wrong results if e.g. newly arriving LCUs require a huge amount of Armament points in one turn but no new LCUs will appear in the following weeks.

Intermingled in all those strategic resources is the display of your current Point Ratio to show you how near you are to auto victory (or loss). This value is actually the ratio multiplied by 100[4], so if the row shows the value of 50 you have half as much points than your opponent, if it is 100 you have both equal points and if it shows 200 you have twice as much points than your opponent. Of course points are neither produced nor used, so the only interesting information besides the value in the table is the development in the history graph.

The tabs on the lower half tell you where the selected resource is **produced,** what it is **used** for (IMHO nice for Naval points) and the last one provides a **history graph** of the storage development to allow a better estimate whether the data shown in the table above is also correct long-term.

The “produced in” sub tab will show factories or resource sources which are currently producing the strategic resource. What is shown is the average daily production rounded to an integer. As factories (not resource sources) produce only 1/30 of their size per day you will often see 0 as amount produced. This does not mean nothing is produced (otherwise nothing would be shown) but just that factory size is below 15. The tab also shows expanding or repairing resources/factories by adding the still to repair value in braces, e.g. “(+5)” means that production in that city will expand by further 5 units per day in the future.

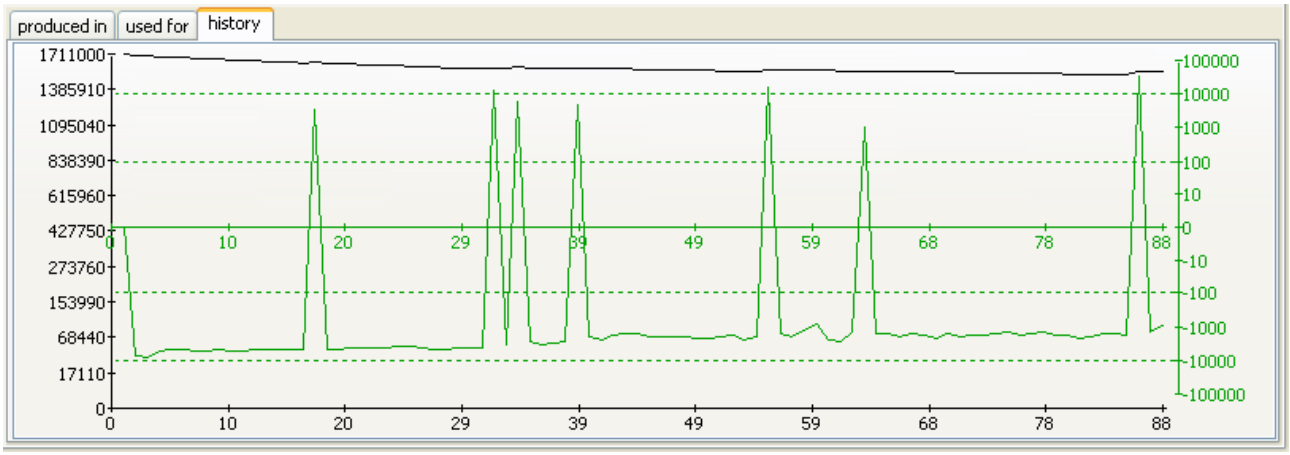
4.1.a The History Graph

The history graph helps to develop a feeling of how storage changes in the long run. Due to the fact that in the long run things can change a lot, the history graph uses a strange scaling of the y axis. Actually it uses two strange scalings.

The screen grab of the history graph below shows the development of the Japanese oil reserves over the first 88 turns of the war. The black line shows the absolute amount of stock available. This uses a square root y-axis meaning that values 4 times as large are shown two times as high[5]. This was done to allow at least a little better resolution for small values than with a linear scale (and a logarithmic scale hid too much of the high value changes). This absolute storage axis start at the bottom of the graph.

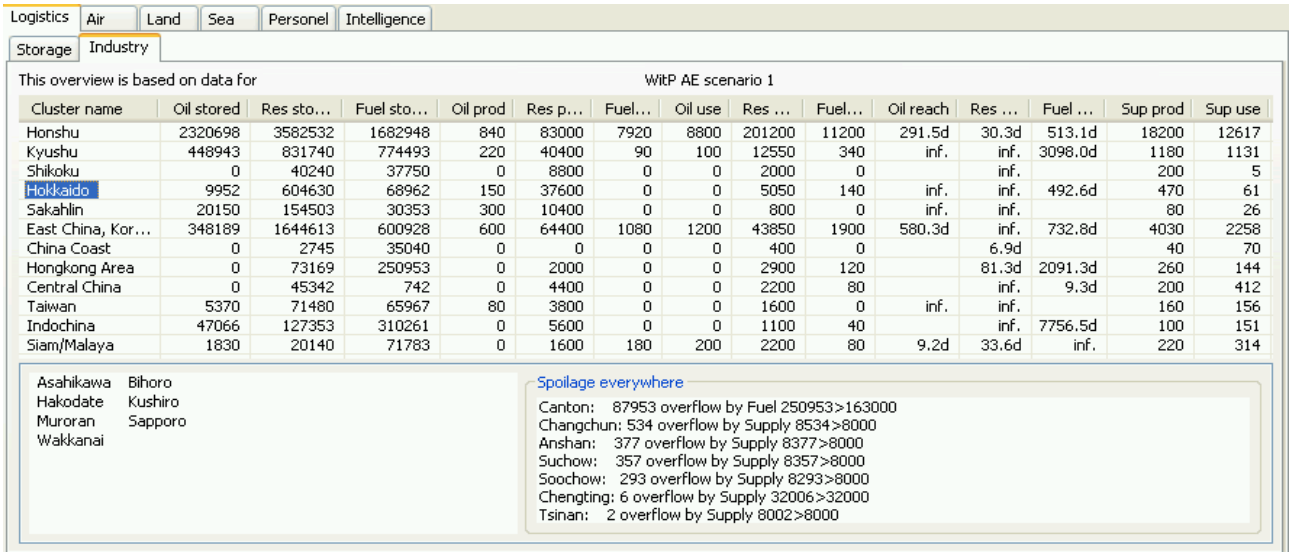
4 Or, to say it otherwise, the percentage of the ratio.

5 Or, in a more relevant way, if the black line reaches the middle of the chart (i.e. the green x-axis), you have used up ¾ of the starting stock, not ½ as it might look when ignoring the axis scale.



The green line shows the relative change to the turn before. The green chart uses basically a log10 scale[6] as can be seen easily on the axis to the right. A log10 scale means that everywhere on the chart values with a relation of 1 to 10 have a constant vertical distance, i.e. on that scale 10 is as far from 100 as 100 is from 1000. This is necessary to show the huge variation in the day to day changes with sufficient detail. What you see in the chart is that in the start Japan had a huge lack of oil production and was using up its resources with nearly 10000 units per day. At turn 88 that lack has been reduced to about 1000 units per day due to various conquests. Also each conquest of a major oil centre brought the stocks at that centre into Japanese possession causing an spike in the green graph. This change axis has its origin not at the bottom of the graph but in the middle (to be able to show negative changes).

4.2 Industry Tab



Cluster name	Oil stored	Res sto...	Fuel sto...	Oil prod	Res p...	Fuel...	Oil use	Res ...	Fuel...	Oil reach	Res ...	Fuel ...	Sup prod	Sup use
Honshu	2320698	3582532	1682948	840	83000	7920	8800	201200	11200	291.5d	30.3d	513.1d	18200	12617
Kyushu	448943	831740	774493	220	40400	90	100	12550	340	inf.	inf.	3098.0d	1180	1131
Shikoku	0	40240	37750	0	8800	0	0	2000	0		inf.		200	5
Hokkaido	9952	604630	68962	150	37600	0	0	5050	140	inf.	inf.	492.6d	470	61
Sakahlin	20150	154503	30353	300	10400	0	0	800	0	inf.	inf.		80	26
East China, Kor...	348189	1644613	600928	600	64400	1080	1200	43850	1900	580.3d	inf.	732.8d	4030	2258
China Coast	0	2745	35040	0	0	0	0	400	0		6.9d		40	70
Hongkong Area	0	73169	250953	0	2000	0	0	2900	120		81.3d	2091.3d	260	144
Central China	0	45342	742	0	4400	0	0	2200	80		inf.	9.3d	200	412
Taiwan	5370	71480	65967	80	3800	0	0	1600	0	inf.	inf.		160	156
Indochina	47066	127353	310261	0	5600	0	0	1100	40		inf.	7756.5d	100	151
Siam/Malaya	1830	20140	71783	0	1600	180	200	2200	80	9.2d	33.6d	inf.	220	314

The industry tab provides an overview of the resources required for **HI and Fuel production per cluster of bases** (i.e. home islands, china, taiwan, ...). Since HI production only works if sufficient fuel and resources are available it is important to keep all HI factories and all refineries well stocked. Oil, fuel and resources will move overland in a limited fashion so not all factory bases have to be supplied directly but only each cluster of bases (i.e. bases connected by good enough roads and railways). Information on each cluster is provided here so that the Japanese player can see where he has to ship oil and resources to keep his production running.

Be aware that WitpStaff has no idea of the current strategical situation on the map so it doesn't

6 Anyone knowing a bit about mathematics will now remark that log10(0) is undefined. That is the reason why the chart only “basically” uses a log10 scale. The area from -10 to 10 uses a linear scale. You will probably not notice that, so just forget it if you don't understand what is written here.

know if all the bases in the cluster actually have an uninterrupted connection between each other and can thus exchange their resources and oil. E.g. if you would attack the Soviets and take Okha in the first turn, its oil storage and production would be counted in the "East China, Korea, Russia" cluster even though its oil could not be used automatically by any factory in that cluster until the Russian lands opposite of it are taken also (on Andrew Brown's map there is an oil pipeline). Until that time, Okha would basically be an own cluster but WitpStaff would add its stored oil to the China cluster. So use with care!

4.2.a ClusterBases.csv

Note that this tab requires two .csv files which are not generated automatically by WitpStaff: “clusters.csv” and “clusterbases.csv”. WitpStaff provides a set of those files for the main scenario. If you use them with any other scenario the values shown in the table may be arbitrarily wrong. You will have to create your own files, with “clusters.csv” containing all the clusters you want to distinguish numbered from 1 upwards (and a “-1” entry showing the scenario the files were made for) and “clusterbases.csv” containing all the baseIDs assigned to each cluster. (the “name” column in the example clusterbases.csv is not evaluated by WitpStaff and only there to allow easier error checking). It is best to edit those files with a spreadsheet program (e.g. Excel or OpenOffice) as those will make it easier to keep the syntax correct. Just remember to save the files again as .csv files.

Currently both .csv files have to be present in the respective database directory you created. Normally WitpStaff will copy the default files from its own directory there. So if you want to use others, you have to copy them over after database initialization. You can change the files afterwards as long as WitpStaff is not running.

If you click on a 'cluster' in the top table of this tab, the area in the lower left corner shows the bases *you currently own* in that cluster[7]. If you see a base name in the lower left table which does not fit to the cluster you clicked at, you are either using the wrong clusterbases.csv file or the .csv file you use has an error.

4.2.b Spoilage

The lower right corner this tab also contains a list of all bases which experience spoilage due to a too high amount of Fuel or Supply stored there. They are ordered according to the amount of superfluous fuel/supply stored. This part of the tab will work without the .csv files mentioned in the previous paragraphs.

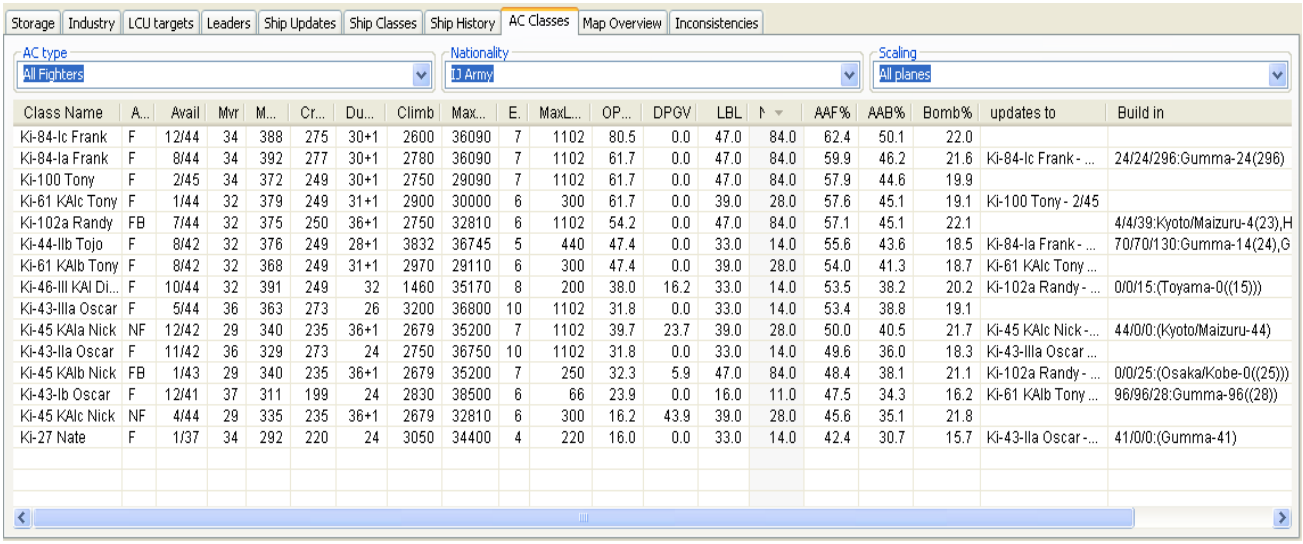
4.3 Aircraft classes

This tab shows the various aircraft classes in the game, their upgrades and their production (for the Japanese). Also plane classes are rated for their expected performance.

The boxes above the table allow to restrict the planes shown in two ways: “AC type” restrict the shown classes to a single type (e.g. Nightfighters), to the main 'supertypes' (“All fighters”, “All bombers”) and not at all (“All planes”). With “Nationality” one can restrict which side or which nationality the planes are from, for the Japanese nationality means e.g. IJN and IJA[8].

7 To make them readable you might have to resize WitpStaff's window, SWT is not really doing there what I want it to do.

8 There is a small problem with shown planes restricted by these 'sub'nationalities: The corresponding information does not exists neither in the save file nor the scenario since plane classes are only assigned to either Japan or Allies but not to individual subnationalities. Thus this information is calculated by using the plane groups existing or entering the game. If a plane groups assigned to the IJA uses planes of a certain aircraft class, that class is assumed to be an IJA class. Also the planes this class upgrades to per default (e.g. Tojo to Frank-a) will be counted as



Class Name	A...	Avail	Mvr	M...	Cr...	Du...	Climb	Max...	E.	MaxL...	OP...	DPGV	LBL	N	AAF%	AAB%	Bomb%	updates to	Build in
Ki-84-Ic Frank	F	12/44	34	388	275	30+1	2600	36090	7	1102	80.5	0.0	47.0	84.0	62.4	50.1	22.0		
Ki-84-Ia Frank	F	8/44	34	392	277	30+1	2780	36090	7	1102	61.7	0.0	47.0	84.0	59.9	46.2	21.6	Ki-84-Ic Frank - ...	24/24/296:Gumma-24(296)
Ki-100 Tony	F	2/45	34	372	249	30+1	2750	29090	7	1102	61.7	0.0	47.0	84.0	57.9	44.6	19.9		
Ki-61 KAIc Tony	F	1/44	32	379	249	31+1	2900	30000	6	300	61.7	0.0	39.0	28.0	57.6	45.1	19.1	Ki-100 Tony - 2/45	
Ki-102a Randy	FB	7/44	32	375	250	36+1	2750	32810	6	1102	54.2	0.0	47.0	84.0	57.1	45.1	22.1		4/4/39:Kyoto/Maizuru-4(23),H
Ki-44-IIb Tojo	F	8/42	32	376	249	28+1	3832	36745	5	440	47.4	0.0	33.0	14.0	55.6	43.6	18.5	Ki-84-Ia Frank - ...	70/70/130:Gumma-14(24),G
Ki-61 KAIb Tony	F	8/42	32	368	249	31+1	2970	29110	6	300	47.4	0.0	39.0	28.0	54.0	41.3	18.7	Ki-61 KAIc Tony ...	
Ki-46-III KAI Di...	F	10/44	32	391	249	32	1460	35170	8	200	38.0	16.2	33.0	14.0	53.5	38.2	20.2	Ki-102a Randy - ...	0/0/15:(Toyama-0((15)))
Ki-43-IIIa Oscar	F	5/44	36	363	273	26	3200	36800	10	1102	31.8	0.0	33.0	14.0	53.4	38.8	19.1		
Ki-45 KAIa Nick	NF	12/42	29	340	235	36+1	2679	35200	7	1102	39.7	23.7	39.0	28.0	50.0	40.5	21.7	Ki-45 KAIc Nick -...	44/0/0:(Kyoto/Maizuru-44)
Ki-43-IIa Oscar	F	11/42	36	329	273	24	2750	36750	10	1102	31.8	0.0	33.0	14.0	49.6	36.0	18.3	Ki-43-IIIa Oscar ...	
Ki-45 KAIb Nick	FB	1/43	29	340	235	36+1	2679	35200	7	250	32.3	5.9	47.0	84.0	48.4	38.1	21.1	Ki-102a Randy - ...	0/0/25:(Osaka/Kobe-0((25)))
Ki-43-Ib Oscar	F	12/41	37	311	199	24	2830	38500	6	66	23.9	0.0	16.0	11.0	47.5	34.3	16.2	Ki-61 KAIb Tony ...	96/96/28:Gumma-96((28))
Ki-45 KAIc Nick	NF	4/44	29	335	235	36+1	2679	32810	6	300	16.2	43.9	39.0	28.0	45.6	35.1	21.8		
Ki-27 Nate	F	1/37	34	292	220	24	3050	34400	4	220	16.0	0.0	33.0	14.0	42.4	30.7	15.7	Ki-43-IIa Oscar -...	41/0/0:(Gumma-41)

Furthermore there is one entry which influences the aircraft performance ratings shown (AAF%, AAB% and Bomb% (see below)). This scaling influences the ratings by taking the best and worst value of each column and using the relative position of the respective plane on this scale as input for the rating. With “current type” only planes of the type selected in the leftmost field are used to span up the scale while with “All fighters”, “All bombers”, or “All planes” the respective other planes classes are taken into account as well.

Most of the column in the table are obvious although size constraints make the headings difficult to read. The non-obvious columns are probably:

Durability: Shows the durability of the plane and its armour if any (after the '+' sign).

OPGV: Contains the “Offensive Penetrating Gun Value” which trys to represent the ability of the plane class to shoot forward. The formula used to calculate it is:
SQRT(effect*accuracy*LOG(range))*(1+penetration)*number_of_forward_guns/2
This shows that the penetration capability of a gun plays an important role (thus the name) in contrast to WitP's numbers which IIRC only take effect and number into account.

DPGV: The “Defensive PGV” is similar to the OPGV except that it contains all guns *not* shooting forward (i.e. those typically used by bombers to defend themselves). The formula used to calculate it is the same.

LBL: Shows the “Land Bomb Load”, a measure which hopefully expresses the usefulness of a bomber against land targets. Its formula is the sum of LOG(effect)*number*4 for all bombing devices. Note that for this measure to achieve a high value many bombs are more important than big bombs.

NBL: The “Naval Bomb Load” is a measure to express how well the bombs (and torpedos) will fare against *war*ships. Due to this penetration is the important factor in the formula and TBs will have much higher values than anyone else: sum (penetration*(LOG10(number) +1)) over all bombing devices.

EAR: The “Estimated Aggregated Research” shows how many 'research planes' of the type were build up to now during the research phase. Every 100 planes supposedly give a chance to

belonging to that nationality. Any aircraft classes further out (i.e. more than one upgrade away from any existing or reinforcing aircraft group) will not be assigned to any subnationality. Also old classes, from which all groups have upgraded away will not be shown by subnationality. Such aircraft classes will only be shown under “All Japanese” resp. “All Allies” or under “All nationalities”.

move the availability date one month earlier so it is interesting to know this number. But note that it is an estimate which assumes that every day the average number of 'research planes' was build.

Build in: This column shows how many planes of an aircraft class are currently build and where. The first three numbers are CurrentBuildPerMonth/ MaxCurrentBuildPerMonth/ RepairingFactories.
Afterwards this is listed for each factory separately, so e.g. “(Osaka/Kobe-0((25)))” means that currently 0 planes can be produced in the Osaka factory, that the factory could be repaired by 25 units and that the factory production is switched off (outermost braces) as well as that repair is switched off (second set of braces around 25). “Osala/Kobe-10(15)” would mean that the factory produces 10 planes per month, that it can be repaired by 15 more units and the production and repair are currently both switched on. At least for me this solves the problem of “which of the starting factories did I convert to build which plane?”.

4.3.a The performance values

The AAF%, AAB% and Bomb% values are intended to be used to compare aircraft against each other (as far as in-game reality is concerned). AAF% expresses the ability to do aerial fighting against other fighters, AAB% aerial fighting against bombers and Bomb% the ability to bomb targets (and get away). These values are calculated using a weighted sum over nearly all the columns to the left of them in the following way.

First for each column the maximum and minimum value is calculated over all the planes according to the “Scaling” selector. So if for example “Scaling” is set to “All Fighters” and the column in question is “Mvr”, the highest and the lowest Manoeuvre rating among all fighter classes is searched for. Then for each plane its Manoeuvre rating is set in relation to this [min;max] interval. If a plane had the maximum value, it would get a 100 rating for further calculation, if it had the minimum value it would get a 0. Then for each plane the values of all columns are weighted and summed up. So basically the formula (with the manoeuvre example) is

```
(maneuverOfPlane-maneuverMin)*100/(maneuverMax-maneuverMin)
*ManeuverWeighting.
```

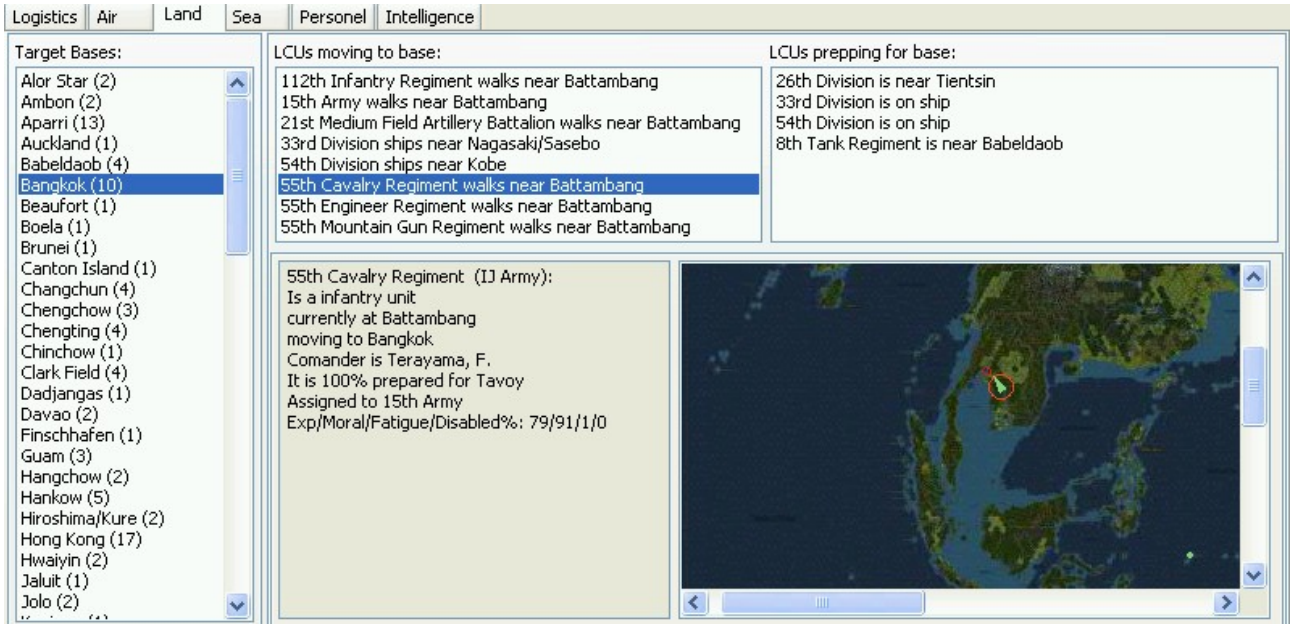
There is one exception to this rule which is durability and armour. Since I feel that armour without durability is useless, both ratings are aggregated into one using

```
(durability*(1+armor*ArmorWeight/100)-duraMin)*100/(duraMax-
duraMin)* DurabilityWeight
```

Thus the weighting of armour only influences how much each point of armour increases durability. So with the default value of 10 an armour of 2 would increase durability of a plane by 20% for the purpose of rating it.

And now the nice part: The weightings can be **adapted** to one's own preferences by editing the “acfactors.csv” file (preferably with a spreadsheet program) while WitpStaff is not running. Note that the last column in that file should (but not must) contain the sum of all other columns except the armour column. But you can also use it as an overall scaling factor.

4.4 LCU targets Tab



This tab tells which LCUs are **moving to a specific base** (either a Japanese or Allies one) or are **preparing** for it and where they are. This should help to see e.g. if indeed a base unit has been put on its way to Wewak or how many units are already shipping to assault Soerabaja. It does *not* show units already at bases (one can easily see that in the game) nor the AV value of the units in question (not yet implemented since no device information at all is currently in the database).

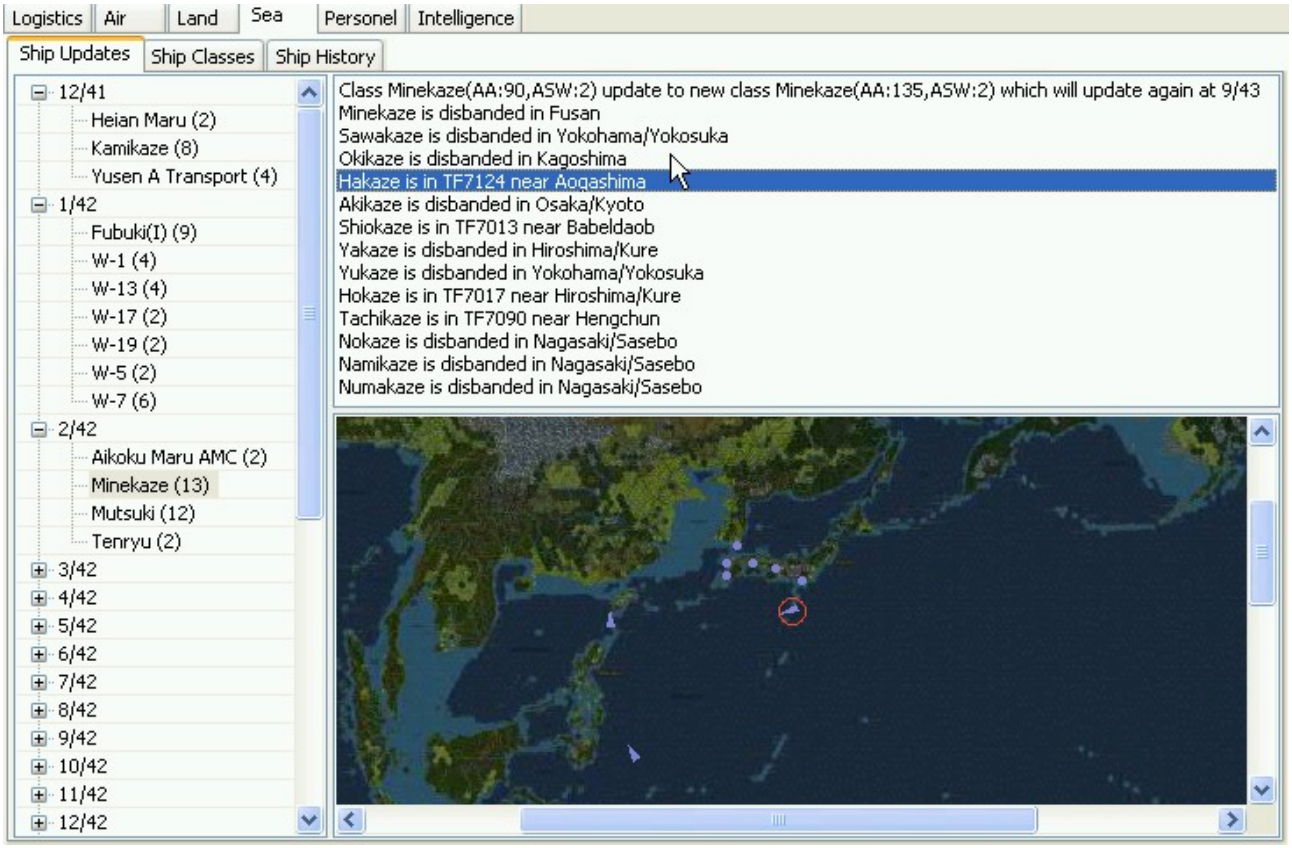
One can get a somewhat more detailed overview of each unit by clicking on it. Also the map will then scroll to the position of the unit. Correct positioning on the map depends on the correct map being used by WitPAE during initialization of WitpStaff. Otherwise units may appear in the correct hex[9], but the map graphics for that hex looks different than it should.

4.5 Ship Updates Tab

For each upgrade month this tab shows all ship classes which will update in that month. The right part then shows per class which ships belong to it and where these ships currently are (also on the map below). So it is no longer necessary to go hunting around in game for all ships that will be eligible for an update in a couple of weeks to bring them home in time. The right part also gives an abbreviated overview of what will happen during the upgrade (changes of AA, DC and Radar). But that can be seen in much more detail in the Ship Classes tab described in the next section.

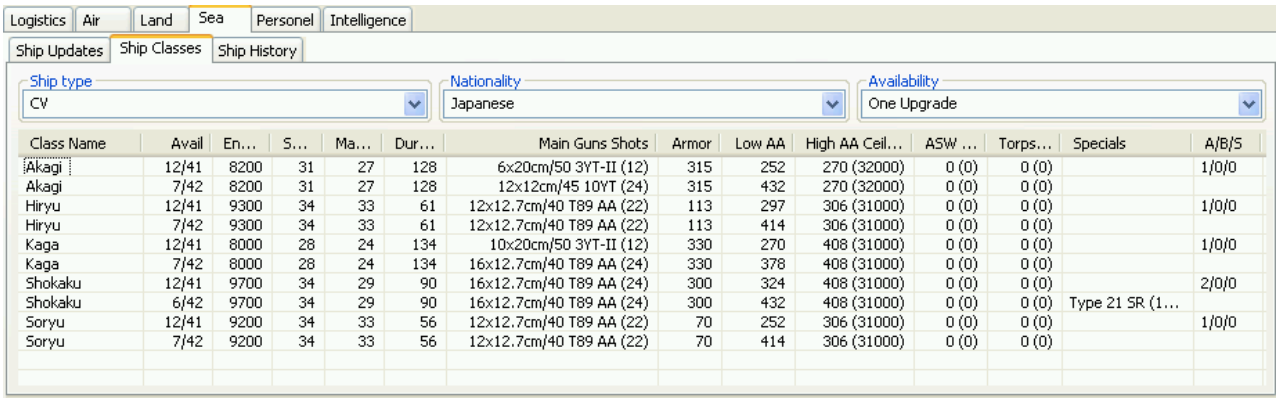
The map will show ships that are disbanded as blue dots and ships that are moving as blue triangles (with the tip pointing approximately into the direction they are moving in). The ship selected in the top right list will be shown with a red circle around it and the map will scroll to it.

9 Since 0.5 everything should be displayed in the correct hex. But don't rely on that.



Note that entries in the left column will show as long as you have non-updated ships if that type. Due to this a ship class can appear under several months, if the ships of that class are in different states of upgrade. Also this means that this tab does *not* contain all updates of a ship class, only the next ones for each ship. For all updates look at the Ship Classes tab. And if a class never updates, it will not be shown in the list to the left.

4.6 Ship Classes Tab



I really liked some Excel sheets send around on the forum some time ago (by DesRon23) which listed Allied and Japanese DDs and Cls with their most interesting attributes. This makes for example the selection of your CVs' escorts easier (long endurance needed). Now WitpStaff contains the same but automatically reads from the scenarios (so it will also work with mods) and with a bit more information. Many (but not all) columns can be used for sorting. The non-obvious columns are from left to right:

Main Guns: These are the guns (Naval or DP) with the highest effect. The number in braces is the number of salvoes these guns can shoot when fully ammunitioned.

Low AA: The aggregated effect halved[10] of all AA and DP guns of the ship class with a ceiling <20000. If you sort for "Low AA" the classes will actually be sorted after *sum of Low AA and High AA*.

High AA & Ceiling: Aggregated effect halved of all AA and DP guns of the ship class with a ceiling >=20000. Also the ceiling of the highest shooting AA/DP guns is shown (the number in braces). If you sort for "High AA" the ship classes will be sorted for this ceiling.

ASW & Rating: Shows the 'official' ASW value (which is simply the number of ASW devices on board) and "WoOS' Special ASW rating", which is the aggregate of effect * accuracy / 100. This rating should reflect the improvements in ASW equipment over the time better than the simple number of devices. When sorting, the rating is used to sort.

Torps & Ammo: The number of torpedo tubes and the *overall number* of torpedoes (so not the number of salvoes as with the main guns). The overall number is used since some ships (or more correctly submarines) carry a number of torpedoes not dividable by their tubes. BTW, the number of torpedoes is stored as a byte in the database, so if you ever design a mod with a ship class with more then 127 torpedoes, your mod will not work with WitpStaff.

Specials: Shows things like radar and rockets.

A/B/S: The number of **a**vailable, **b**uilding and **s**unk ships of this class in the game.

The three Combo boxes at the top of the ship class list allow different restrictions on what ships are shown: by ship type, by the side that owns the class, and by the number of future updates shown.

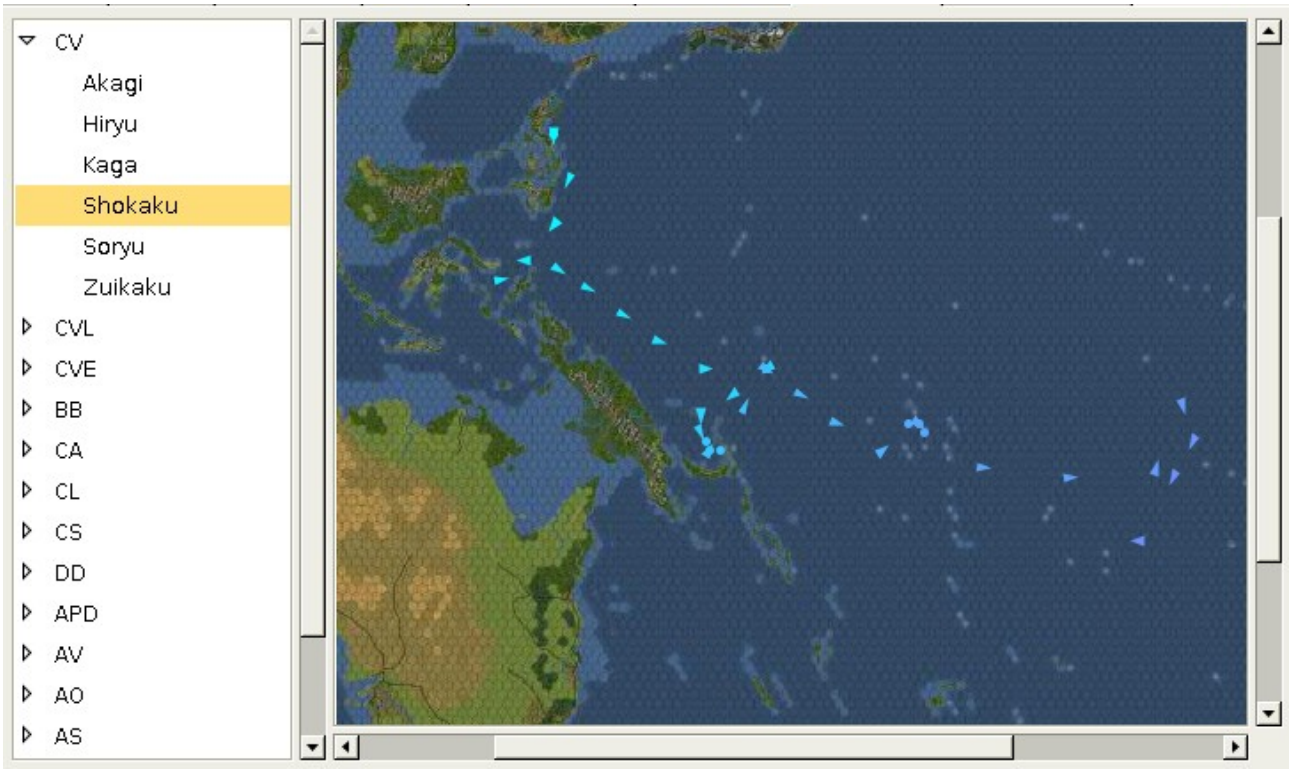
If you *double* click on a class name (single click will not work) you will be transferred to the "Ship Updates Tab" (see chapter 4.5) where the position of all ships belonging to this class is shown on the map. For this operation "belonging to this class" means "belong to a class which has the same name as the one clicked on". If you use a scenario where classes change names between versions, the results you get from double clicking might not be what you expected.

4.7 Ship History Tab

The ship history tab shows where each of your military ships[11] has been within the last 512 turns (or fewer, depending on how many turns you have read into the database). Below for example one can see that Shokaku first supported the cleanup in the Philipinean sea, then help with the conquest of Rabaul, rested a bit in Kwajalein and then went high speed west to intercept the Allied CVs which had just raided Canton Island and sunk parts of Japanese the invasion fleet. As you can see this might allow you to remember old victories, might help with AARs or might help with analysing AI behaviour.

10 Don't ask me "why halved?". I'm simply replicating the data shown by WitP.

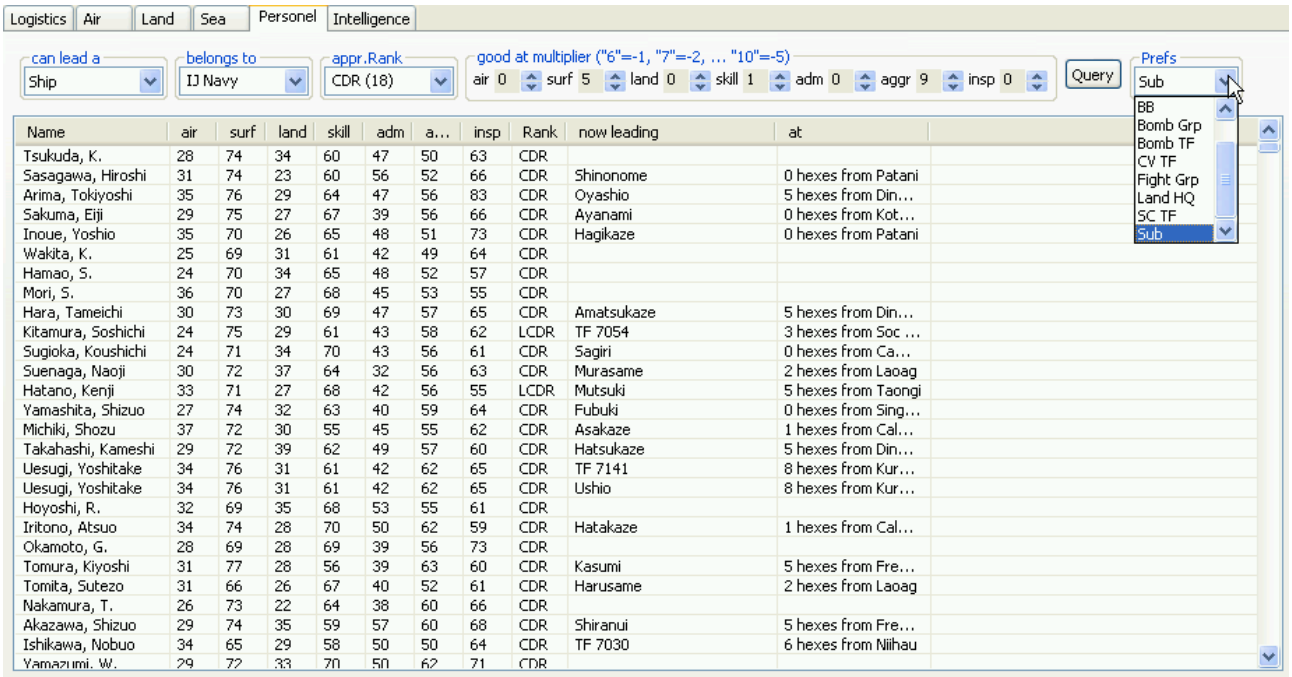
11 Actually you can influence for which ship classes data is collected. Edit the ships4history.csv file *before* initializing a new database. I would suggest against tracking APs and AKs due to the storage requirements involved.



If the map shows an arrow, the ship moved approximately in that direction. If the map shows a dot, the ship was stationary. To better show how long a ship stayed at a fixed position (most probably in a harbour) these dots are spread around a bit even if the ship doesn't move (can be seen north of Rabaul and in Kwajalein the above picture).

4.8 Leaders Tab

This tab provides an overview over **all leaders** (even those already assigned) showing all their abilities (without the need to click on them) and with the ability to sort after useful, user-defined criteria.



To avoid long query times it is very much recommended to restrict the leaders looked at with the "can lead a", "belongs to" and "approximate rank" conditions. The "good at multipliers" allow you to define your own rating system. Note that values over 5 actually represent a negative influence[12] of that attribute on the rating. So to search for a submarine captain you would select:

can lead a: ship

belongs to: IJ Navy

air: 0	=multiplier 0	
surf: 5	=multiplier 5	(we really want to have that)
skill: 1	=multiplier 1	(maybe nice for training)
agr: 9	=multiplier -4	(and high aggression should be avoided)

So e.g. Tsukuda from the picture above has a rating value of 5*74+60-4*50=230. This rating will order leaders with high surface skill, some overall skill and a somewhat low aggressiveness to the top. And now go through your save files and have a look whom you actually did select ;-)

The "Now leading" column shows which unit/ship/TFs the leader is currently assigned to. This helps to find free leaders or rearrange leaders from non-fitting assignments. If there is a "(NYA)" after the name of the unit in this column, that unit has not yet appeared on the battlefield. The leader assigned to this unit can not be used elsewhere and can not be reassigned until the unit appears.

Note that the leader information from the save games seems to be a bit unreliable, e.g. leaders are commanding kinds of units in-game they cannot according to the "can lead a" entry in the save game. Also it seems that not all leaders shown in this tab will always be available in-game when assigning leaders.

4.8.a Preferred queries and leader-prefs.csv

The "Prefs" button to the right allows for standard queries to be defined , so less clicking on buttons is necessary. You can define your own preferences by editing the "leader-prefs.csv" file (preferably with a spreadsheet program). Its contents should be rather self-explanatory just keep in mind that the texts used for the "can lead a" and "belongs to" column must exactly match the texts you see in the selection boxes in WitpStaff and that "rank" is expressed as the number the rank has (see at end of rank abbreviation in braces).

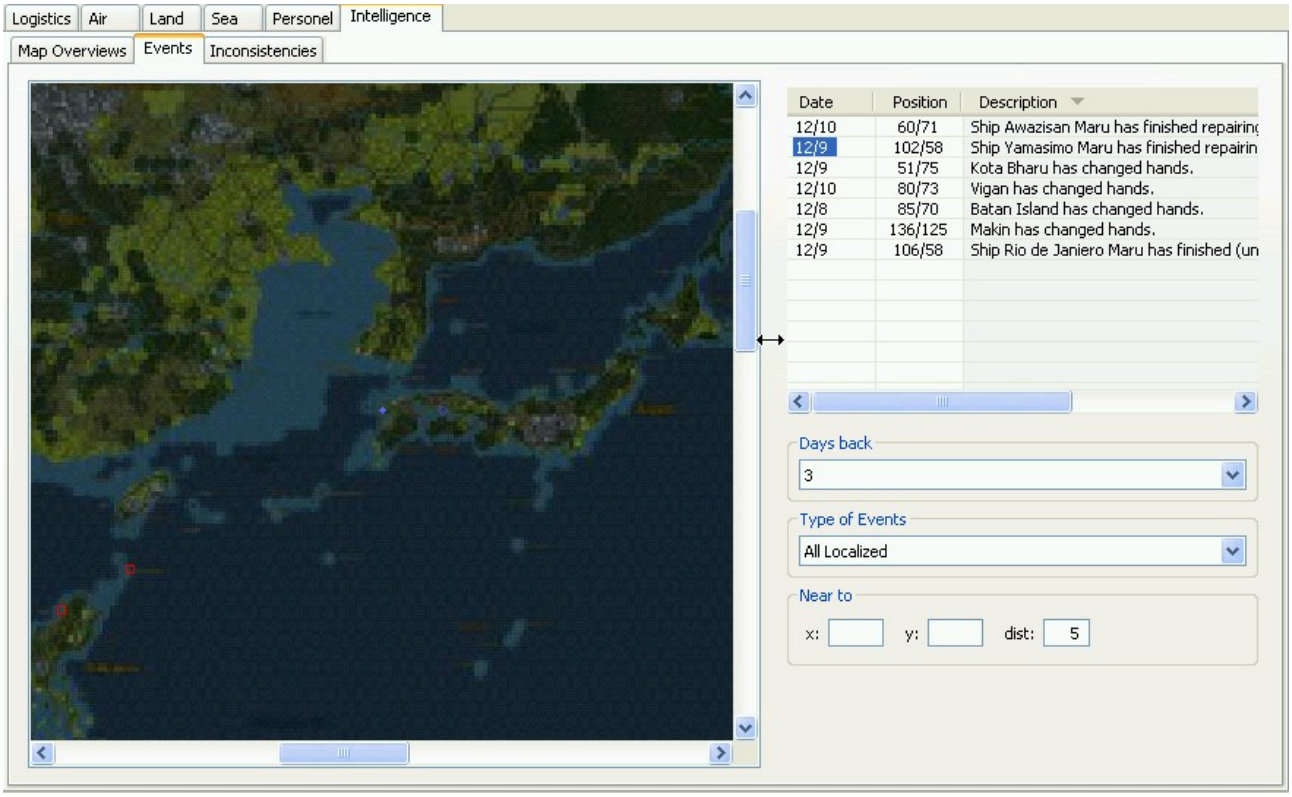
4.9 Map Overviews Tab

This tab is so configurable and content rich that it got an own chapter. See below.

4.10 Events Tab

The Events tab shows things which happened in turn resolution, such as occupation of bases, newly arrived ships, or changes in the skills of leaders. In the end it is intended to also be able to show data excerpted from the turn actions files, but that still requires some work.

12 Alas SWT does not allow spinner buttons to go into negative values, so this workaround was necessary.



In the left window the Events tab shows the map with icons for all the selected events. On the right upper side, the events shown on the map will also be listed in text. On the right lower side, one can select which events should be shown on the map and in the list. There are three ways to restrict the data shown:

Days back: How many turn resolutions back shall events be shown.

Type of Events: The types will be mostly retrieved directly from the database but there are two special entries: “All” will show all events and “All localized” will show all events which have a position > (0,0).

Near to: Allows to set a center point and define the radius around it to show events. Clicking on the map to the left will automatically enter the hex clicked on with an radius of 5. To remove that restriction clear at least one of the three 'Near to:' fields.

Future versions will contain an extensible plug-in which will allow also to read events from the action log files of the turn resolution into WitpStaff's database.

4.11 Inconsistencies Tab

Any inconsistencies found in the save file (for which the built-in consistency maintenance of the SQL database had to be switched off or circumvented) will be shown here. Currently this includes leaders leading several TFs/units, units with no or a foreign leader, bases in the clusterbases.csv file which don't exist in game, LCUs with wrong suffixes or HQs, ships assigned to non-existent TFs (actually caused by TFs claiming to be of nationality 0), and TFs with non-existent home ports.

Note that for implementation reasons leaders which command *several TFs and a ship* will show up as inconsistent mentioning the ship in addition to the TFs although captaining the ship does not belong to the inconsistency (one can lead a ship and one(!) TF).

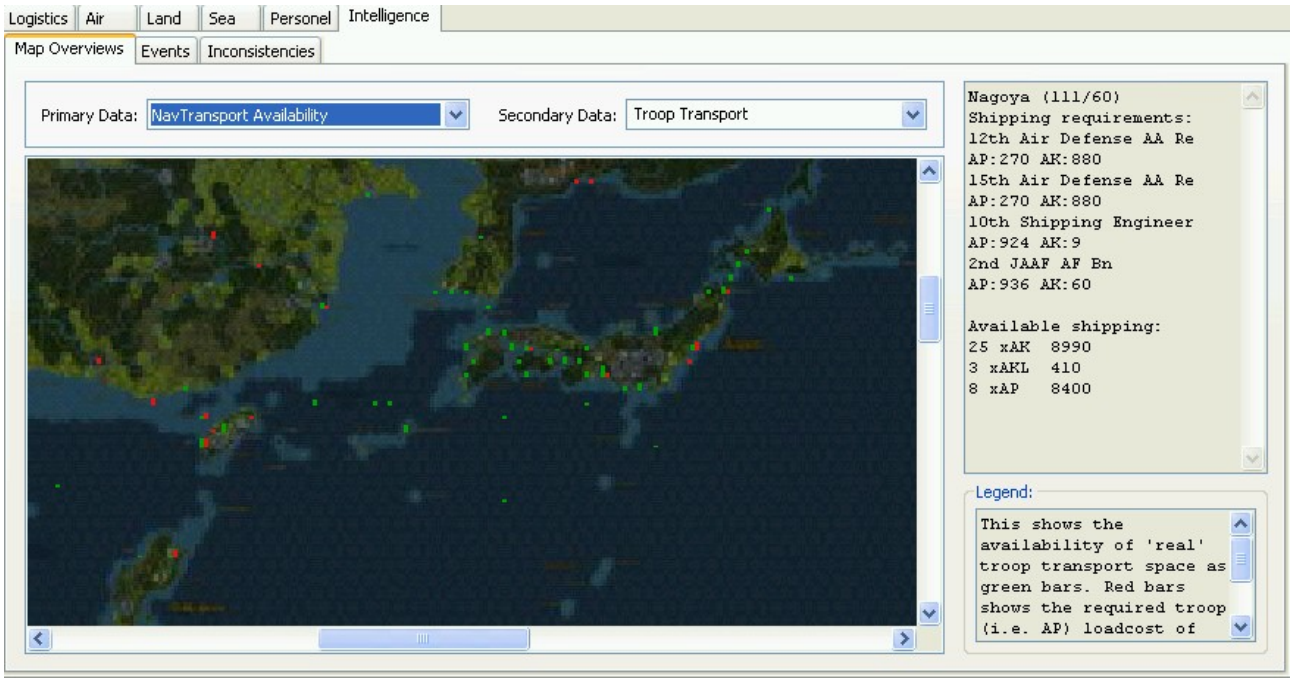
If you switch on the selection box on the top you will also be shown inconsistencies involving future leader or units. This is mostly interesting for scenario designers as the normal player typically

can not do anything to resolve such future inconsistencies.

5 Map Overviews

Lots of information in WitP requires the context of the map to make it more accessible. This is the domain of the map overviews of which WitpStaff currently contains four: Naval transport availability, Naval search coverage, Support situation, Supply situation, and attack power concentration of LCUs. “Plug-ins” for map overviews can be written by users in two variants. The newer variant allows for one plug-in to show several related displays (via the “secondary data” mechanism) and to display some explanations in the “Legend:” window.

5.1 Naval transport

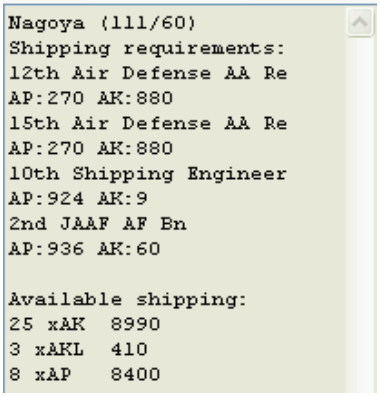


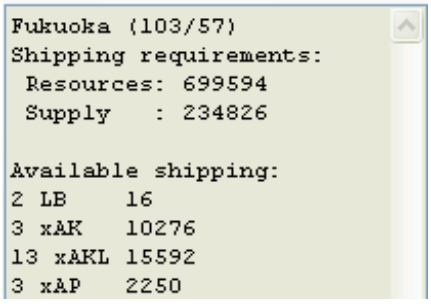
This mode shows the availability of naval transport capacity and the required amounts of it. The green bars will in general be the available capacity in a harbour, the red bars the amount of things to transport lying (or sitting, drinking, ...) around in the harbour. Sometimes it is difficult to determine whether things (typically LCUs) really want to be transported.

Some more detailed explanations are given in the “Legend” window.

5.1.a Side column

Clicking on any one of the hexes with a bar in it will produce some data on that hex in the side bars:



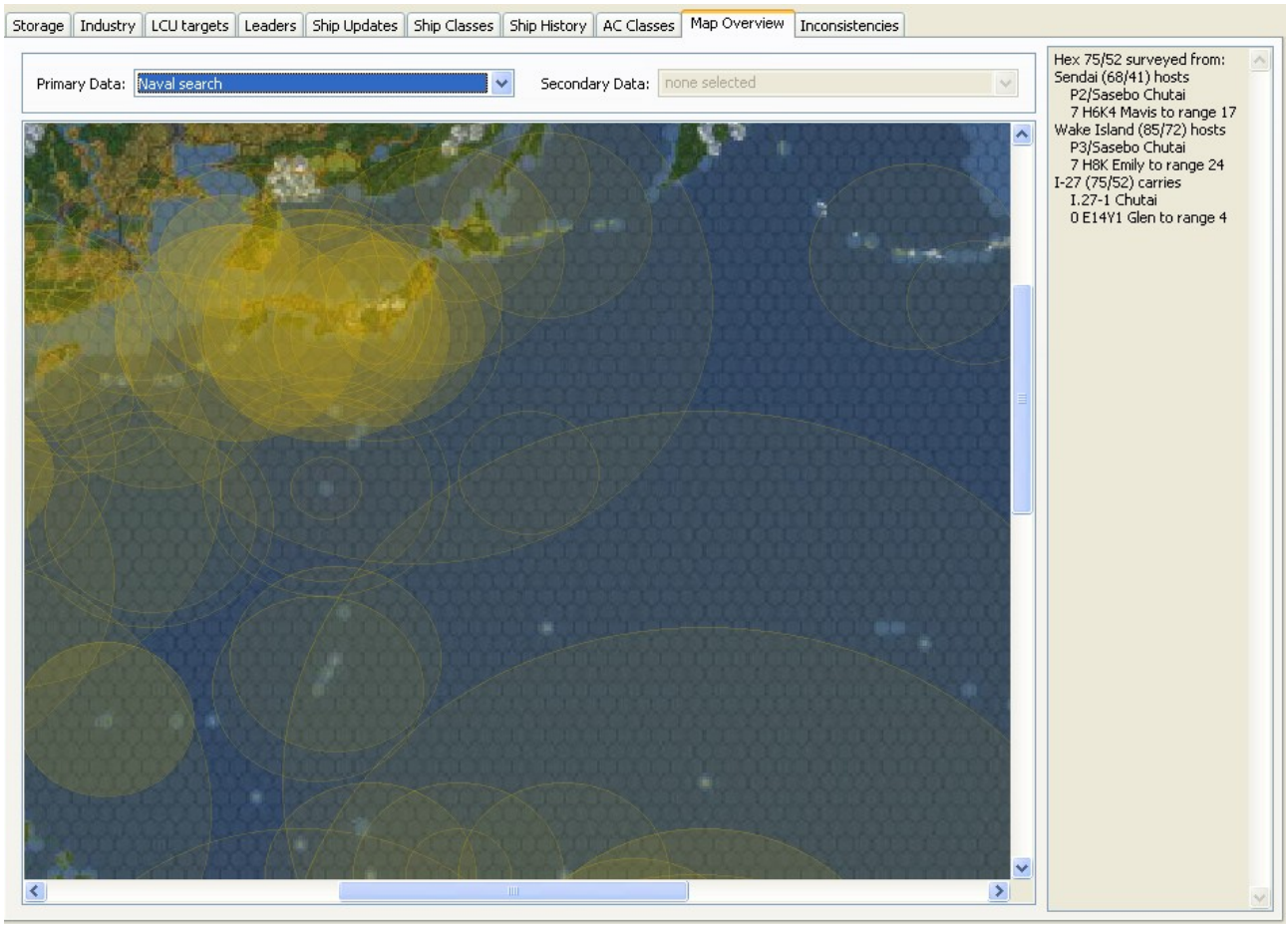


The left picture shows the information for Nagoya in the “Troop Transport” mode. It lists at the top the LCUs in Nagoya which prepare for a base other than Nagoya. For each of them the shipping requirement in personel (AP) and in bulk points (AK) is given. Remember that an LCU requires *both*, so if loaded onto something which does only provides one of the two storage facilities, the rest will be crossloaded using up more than the shown capacity.

Below the LCUs the available shipping capacity in Nagoya is shown.

The right pictures shows the status of Fukuoka in the “Bulk Transport” mode. The top shows what is laying around in the harbour (although it will probably not need to be shipped away this time) and the bottom the available shipping capacity.

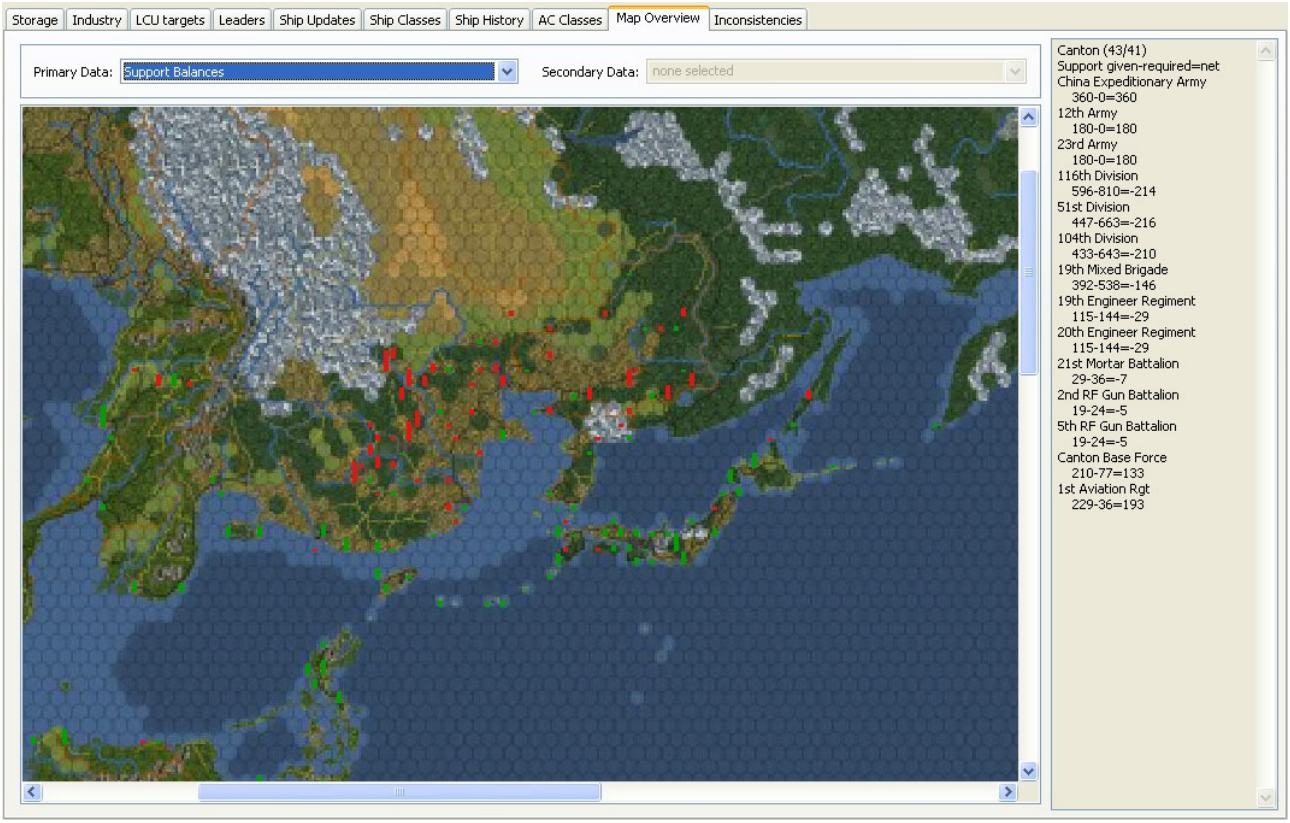
5.2 Naval search



The Naval Search view in the Map overview tab hints at the coverage achieved by naval search and ASW planes. Search areas are painted orange with different amounts of transparency. The display takes into accounts the number of planes actually flying (i.e. size of group*search %) for determining the transparency and the range for determining the area to paint (according to some research by Apollo range does not influence search efficiency). Both LBA and ship based planes are included. Overlapping circles add up their opaqueness according to an unknown formula (whatever SWT has implemented). Note that there is a minimum opaqueness (ca. 1%) and a maximum one (ca. 20%) per circle. The latter is typically reached with 50 planes searching 100%

The sidebar shows for each hex clicked at which planes search this hex and from which airport or ship they start. Since number of planes is multiplied with the search% a quantity of 0 is possible although at some days a plane will fly.

5.3 Support

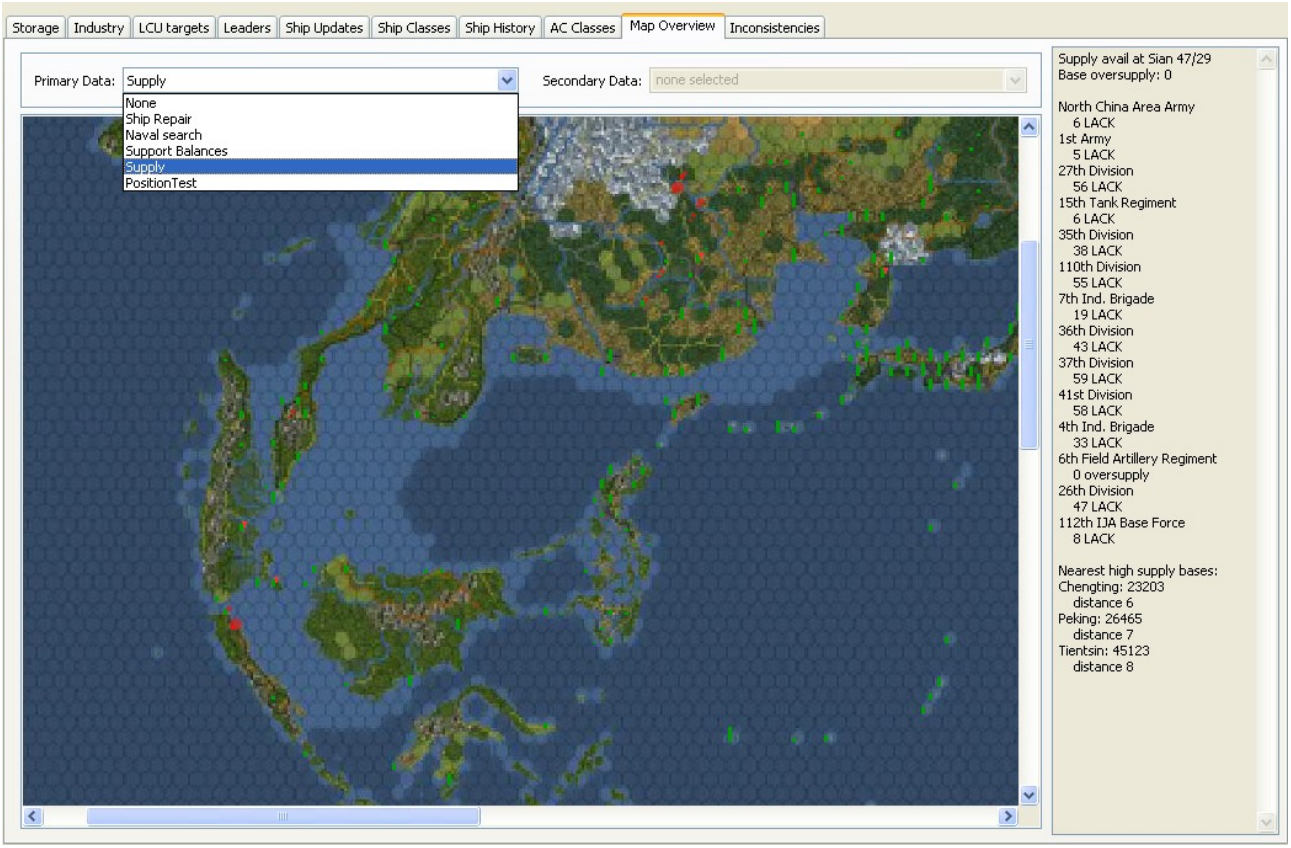


The support view shows whether your troops are receiving enough support in each hex (independent of base or no base). If the column shown is green, enough support troops are in the hex. If too few support is available, the column is red. The length of the column indicates the amount of the problem but using a quadratic scale (so that also small amount of lacking support will be visible). 15 support squads too much/too few will fill 1/4 hex, 60 support will fill 1/2 hex, 230 support too much/too few will fill about 1 hex, 500 support surplus/lack will fill 1.5 hexes and more is cut off.

Note that due to some strange implementation in WitP (even 1 support too few will cause continuous loss of support troops to disruption) it is advisable to have as much troops as possible in green hexes.

The sidebar on the right shows the number of support squads and support requiring squads in the clicked on hex. Note that some squads neither provide nor require support (e.g. Aircraft support) and are thus not shown here.

5.4 Supply



This mode of the overview shows where supply is available in surplus and where it is lacking. Basically it evaluates all the "supply required" fields of bases and troops and compares them against the available supply The overall difference between available and required is then shown as a green (=surplus) or red (=lack) bar in the hex. Similar to the "Support" mode of the map overview the scale of the bars is quadratic, with 100,000 supply surplus being 1 ½ hexes high, and 10,000 supply lack being also 1 ½ hexes high[13]. This translates into ca. 45,000 supply surplus being one hex high, ca. 11,000 supply surplus being ½ hex high and ca. 2,500 supply surplus being ¼ hex high.

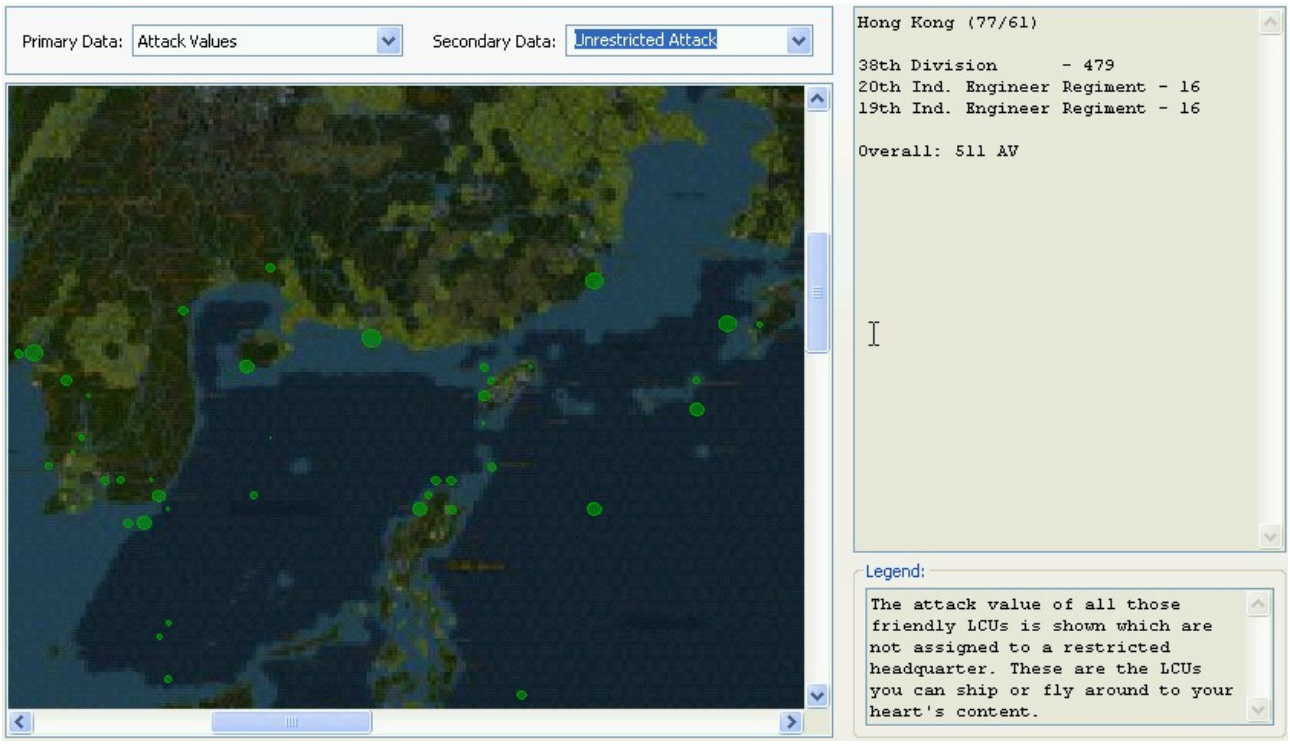
Since a base can have an overall surplus despite some units suffering a supply lack (seems to always happen in Rabaul) the hex will show a dark red circle of different size if some troops in the hex suffer from a lack of supply. Size of the circle is proportional to the number of units involved.

Also a hex may have too many supply and suffering spoilage. This is shown by a light red triangle (supposed to represent a warning sign).

The right sidebar shows for each clicked at hex the supply situation of the base (if the hex has one) and all the units in the hex. If either base or at least one unit has a lack of supply it will also show the 3 nearest bases with a huge (=more than 20000) surplus of supply. Note that "nearest" simply means flight distance and does not take any obstacles into account.

5.5 Attack value

13 Note the different number of zeros in both numbers. Lack of supply is shown bigger than surplus of supply since lack is often much smaller.



5.6 Other Plugins

The Overview tab has been rewritten from the V0.4c version (where it first appeared) to use plugins for each of its modes. The idea here is that users can write their own plugins (knowledge of Java and SQL required) and have WitpStaff automatically integrate it. For this several interfaces have been designed which will be described in a separate document (to avoid shocking the laymen ;-). Look in the de/retsiemuab/witpDecoder/gui/mapviewPlugins folder.

6 Warnings

- Nothing is completely tested. Don't complain to me if you think WitpStaff lead to the loss of a battle, a base or the whole war.
- Since each version up to now introduced new features there are also new bugs to be expected.

7 Known Bugs

The following bugs are known but difficult to fix.

- WitpStaff has problems with the fact that in WitpAE not only planes but also engines can be researched. The Storage tab will show production for engines which are still in the research phase and thus are *not* produced. Needs some rewrites of the code for resource handling.

8 Extensions

For obvious reasons the source code of this tool can not be released. Thus this is not an Open Source program!

The Map Overview Tab has been (re)written in such a way as to allow 'easy' adding of new view modes via “PlugIns”. See section 5.6, a yet to be written documen, and the example classes.

The map actually used in the Map overview (and in other tabs) can actually be hand-modified. Currently it is an automatically blown up version of a map used by WitPAE for its overview screens . This results in limited quality but the ability to adapt to any map modification applied to WitPAE. You can create your own map by replacing “MainMap.jpg” and “BigMainMap.jpg” in the respective database directory with a version covering the same area. The sizes of the images can vary, WitpStaff will automatically adapt itself (within limits, of course). You might also want to check with the “Position Check” test plugin of the Map Overview Tab whether you correctly cut the map for the hex placement to work.

If you think there is other functionality missing, you can always help me to add it by providing the code for an appropriate tab. Each tab is a SWT Composite object. It gets called in the beginning to construct it and to assign a related GUI Adaptor class, which is then responsible for initializing the content and updating it according to GUI events sent to it. The adaptor is passed a JDBC database object allowing to query() the SQL database. The DB schema can be seen e.g. from the witp.script file (but I can of course explain a bit more than that file says).

If you are interested, mail/PM me and I can send you more information (probably including example classes). If you don't understand what the paragraph above says, don't contact me. I'm not going to give courses in DB or SWT programming for you.

9 Copyright & Co.

Ah, very simple:

1. You are free to use the program.
2. You are not free to try to re-engineer the code for parsing the save file. Doing so voids your license to use the program.
3. There's no guarantee. None. Really. Consider, it is Java. Java is not from Microsoft. And it contains Open Source software. Boy, must it be buggy.

10 Thanks

Without the HsqlDB database system this program would have been much more complicated to write. Using HsqlDB I could delegate most of the tedious search and sorting tasks to the database. Similarly without SWT the tool wouldn't look so nice on the Windows platform (it doesn't really look like Java, does it?). Eclipse provided the IDE, Jigloo the GUI editor for the first few screens, which alas crashes as often as the native one of Eclipse (thus actually nowadays I use the latter).

Also thanks to some of the forumites of the WitP forums which took over the task of explaining this program to those of you who did not read the manual to the bitter end (because I don't really see the value in explaining things again I already spent time on explaining in the manual), provided clusterbases.csv files, plugins, ...

And thanks finally go to Matrixgames without which this version of (formerly) WitpDecoder for AE would not exists, as I certainly wouldn't have reverse engineered the save file format again. Thankfully this time I got the .h files ;-)