CaptainCasa Enterprise Client

CaptainCasa & the SOA Gateway

Modernizing Legacy Applications

Table of Contents

CaptainCasa & the SOA Gateway

There are a lot of applications out and productively running within your enterprise. In most cases projects are not starting from scratch, but need to take into consideration that existing business applications need to be integrated into an updated processing in front of.

A lot of these applications do not provide an up to date user interface, but provide character based terminal user interface (such as the 3270 interface). This makes it difficult, to preserve the value of these applications into the next phase of their usage.

As a result, modernizing the front end is an essential part of securing the investments that were and are made in existing applications.

User Interface Considerations

Performance first!

There are a lot of negative comment about character based terminal user interfaces: difficult to use (esp. for beginners), old-fashioned in terms of look and feel, etc.

But, on the other hand, experienced users have fallen in love with them:

- Good performance
- 100% keyboard driven, if required

Especially the performance aspect is a crucial one – legacy applications typically manage a lot of data input. They represent the back office applications and are the "working horses" of an enterprise.

Consequently, when thinking about modernizing the user interface to these applications, performance considerations need to be addressed from the beginning. - Performance is the key to usability.

Low Bandwidth Connection between Client and Server

System administrators like terminal based applications because they provide a very predictable behavior, when it comes to measuring the data volume exchanged between terminal client and server.

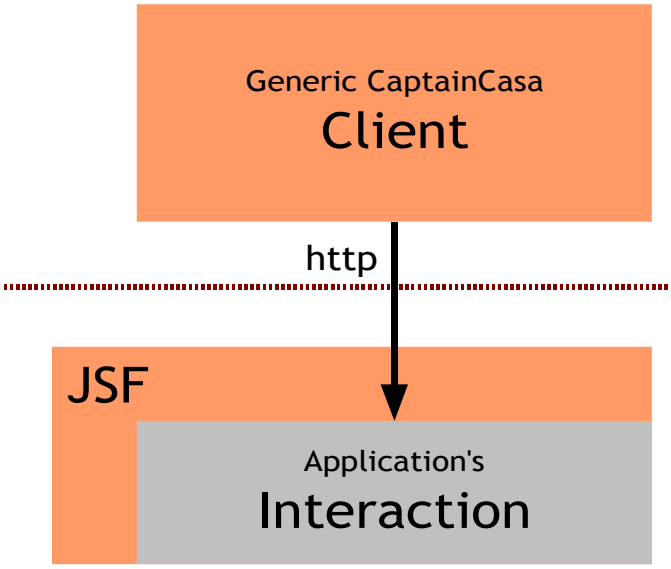
Transferring an application into an up to date user interface must not mean a dramatic increase of this data volume. Legacy applications typically run in central computer centers – with clients all over the world accessing them. Aside the efficiency and cost-savings behind optimizing the data volume exchanged between client and server, there are still many countries worldwide that do not provide adequate network capabilities for exchanging high volumes of data – so smart usage of bandwidth is an important point for a worldwide roll-out of front ends.

CaptainCasa Enterprise Client

CaptainCasa Enterprise Client is a rich client infrastructure that exactly meets the requirements of applications that are operationally used: high client performance, nice look and feel, many high quality user interface components out of the box, low bandwidth usage and a simple server side, Java based connection to existing processing.

In fact CaptainCasa Enterprise Client is built on the basis of a modern terminal protocol, so it preserves all the good experience of existing applications on the basis of an up to

date user interface client.



CaptainCasa Enterprise Client is based on J2EE standards and as consequence is built on a standardized, scalable infrastructure that runs in thousands of applications already.

Legacy Integration Considerations

The user interface processing needs to somehow integrate into the legacy application. And that's, when problems start. Even though a legacy application might be structured in a nice way, exposing properly formed functions, it's still a tough technical challenge to effectively manage the communication between the user interface processing and the legacy application.

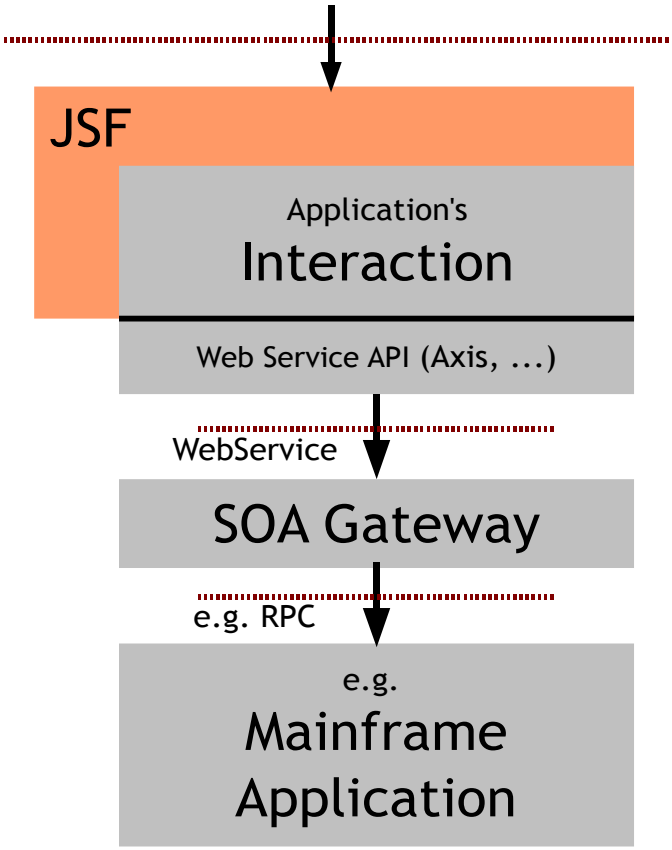
And that's when SOA Gateway comes into play.

SOA Gateway

The core idea of SOA Gateway is simple: Let's use **the** data transfer technology in order to talk to legacy applications that is standardized throughout all operating systems in the meantime: “web services”!

Standardized Communication to Legacy

Using web services means, that legacy functions that are exposed in proprietary, complex formats are made available in a standardized format that is call-able by all up to date programming environments. Imagine SOA gateway as an engine, to which you can connect by calling web services, internally transferring the web service calls into the difficult to use legacy communication protocols.



SOA Gateway hides the technical complexity of, for example, connecting to mainframe based legacy applications, so developers in front of this application are unburdened from this complexity. Developers can concentrate on their job, such as providing nice frontends, and can leave the ugly technical stuff to SOA Gateway.

From Legacy to Web Services

SOA Gateway provides a large number of technical interfaces in order to connect to legacy applications, including:

- Interfaces to existing applications written in languages such as COBOL or Natural.
- Interfaces to existing databases including relational databases like DB2, Oracle and MS SQL Server, and non relational such as VSAM.
- Interfaces to existing environments like CICS on z/OS.
- ...and so on

These interfaces are provided on many operating system platforms – including mainframe and UNIX. - Whereas a number of providers of communication bridges to, for example, mainframe legacy systems need to install explicit servers in front of the legacy system, SOA Gateway is directly installed within your legacy environment. This means a significant reduction of complexity, both from an architectural and from a system administration point of view.

Development Process

On top of its communication infrastructure SOA Gateway provides a development environment, that allows users to quickly make legacy application functions available as web services to other systems. The typical steps that one needs to take in order to

provide these functions are:

- The SOA Gateway development environment presents a list of available structures (e.g. procedure calls) for the legacy application.
- By simple selection and configuration corresponding web service definitions (WSDL files) and web service endpoints are created – without any need of programming.
- These web services can be immediately used and tested.

Once installed and configured, the transferring of legacy structures into web service format is a simple to use, efficient process – not only from technical perspective, but also from communication perspective: legacy system developers expose their knowledge in a format that is standardized and simple to use for developers accessing these functions.

Contract based Development

When one provides functions that are used by others, then it makes sense to define a "contract layer" between both of the parties. This exactly is what SOA Gateway provides: the legacy system is made available by exposing a series of WSDL definition (web service definition language), each of them describing certain call-able operations with their input and output data format.

Because this WSDL definition is standardized there are many tools and frameworks available in nearly all programming environments to access these operations. The WSDL serves as contract between the function provider (legacy system) and the program accessing the function.

Integration Scenarios CaptainCasa – SOA Gateway

Server driven Integration

CaptainCasa Enterprise Client and SOA Gateway meet on the server side. Because of its terminal based architecture, the frontend development of CaptainCasa is done on the server side – so it is not the end-user client that directly calls web services to the legacy functions!

This has many advantages:

- The web services that are exposed by the legacy system through SOA Gateway are called within the server environment and do not need to be exposed on a more public level. This means a significant reduction in security administration and risk. - Of course you can still expose these functions on a more public level, you just do not need to have to.
- Round trips between the user interface layer and the legacy systems are executed on server side, "within the same computer center". This again means a drastic reduction of complexity when it comes to data volume, round trip and bandwidth calculations.

CaptainCasa Enterprise Client provides dedicated functions to simply import web service definitions and integrate them into the user interface implementation. It's the counter part of what happens within SOA Gateway: while SOA Gateway exports legacy functions in a simple way as web services, CaptainCasa imports these web services and makes them available as simple Java API for frontend development purposes.

Pure Frontend Modernization Scenarios

CaptainCasa Enterprise Client and SOA Gateway are designed to effectively support scenarios in which legacy applications should receive an up to date frontend.

- The legacy system is transferred step by step into a “business processing engine” providing functions as web services to outside programs.
- The frontend is able to cope with performance requirements of applications that are operationally used.

A clear contract layer between both systems is the basis for efficient development on both sides.

Business Integration Scenarios

In many scenarios there is the necessity as well to integrate additional functions into the server side application processing. Rather than integrating these functions directly into the legacy application, it often makes sense to define an integration layer in front of the legacy system to “orchestrate” these functions.

Dependent on the complexity of these scenarios, the integration technology may range from simple Java programs, calling a certain sequence of web services up to complex integration scenarios, in which you may use certain integration platforms for orchestrating the calling of diverse services.

Summary

Though – or maybe: because of - being individual products, CaptainCasa Enterprise Client and SOA Gateway form an efficient way to provide modernized front ends for existing legacy applications.

Due to the fact that this modernization is not done on base of some “magic” and “hidden” technology, but is done on clean contract definitions and on a clear separation of functions between the legacy processing and the frontend processing, the value that is created is much higher.

On SOA Gateway

“SOA Gateway” is developed and marketed by Risaris Ltd., Ireland. - Risaris Ltd. and CaptainCasa GmbH are technology partners with a close development relationship.

More information about Risaris Ltd- and the SOA Gateway are provided via the following links:

- http://www.risaris.com
- http://www.soagateway.com

CaptainCasa GmbH

Hindemithweg 13
69245 Bammental

Tel +49 6223 484147

http://www.CaptainCasa.com
info@CaptainCasa.com