4 Classification of Internet Standards and Technologies

Technologies are the third dimension in the Web application framework architecture introduced in Sect. 2.2. The need to consider technologies throughout the process of Web application design is motivated in Sect. 2.2.3. The architecture developed as a result of the stepwise approach described in the previous chapter is not tied to any concrete technology and is free of the specifics of any module of the framework architecture. Now we have to connect the neutral architectures for the Web application and the platform to concrete technologies in order to determine how to implement them.

In this chapter we introduce an exemplary classification and discuss some principles as to how it is constructed and how it can be changed (Sect. 4.1). In this we continue to develop the stepwise approach begun in the previous chapter by shaping the technology-free architecture through assigning technologies to the WAA components and WPA modules in each matrix quadrant.

Almost any of the Web application design approaches, some of which were presented in the previous chapter, have some kind of classification. Some of these approaches formulate it explicitly (our approach, the UML approach), others assume it implicitly (pole shoe notation). The classification introduced in this chapter is no exception to this rule, when it comes to deriving a number of concepts such as logic or presentation, which will be further used to characterize the WAA modules. What makes the role of the classification unique is that it boils down to concrete technologies. If designers go about assigning these technologies smartly, they will easily discover that the classification gives them a tool for the comparative evaluation of different application designs.

It is important to point out that the proposed classification is simply one of many alternatives. Myriad technologies and standards that can and are used in the process of developing Web applications exist. It is an illusion to expect that a thorough classification can be prepared and agreed upon broadly. The readers are encouraged to view our "proposal" critically, to develop it further, and to extend it by tailoring it to their individual needs. The lack of absolute precision of our classification does not impair the design approach. Without doubt, it fulfills our overall goal to provide some structure in the jungle of Internet standards and technologies as depicted in Fig. 2.1.

4.1 Classification

The proper choice of the root concepts is crucial to any classification. The packages of the WAA (Sect. 2.5) are chosen as classification roots (Fig. 4.1). This is a choice which is empirically made and is difficult to motivate. The authors' practical experience shows that WAA packages are a plausible choice which can be successfully used when building Web applications.

The classification involves multiple trees starting from the root packages. Either abstract categories or category bags (CBs) are used to classify further. An abstract category is (consider for example category 1.3 in Fig. 4.3) a category to which no technologies are directly assigned; rather they contain further category bags. Category bags contain different alternative technologies or standards (consider for example CB 1.3.1 in Fig. 4.3). The goal pursued with this taxonomy is not to classify all outstanding standards and tech-

nologies. It is to have a classification which is detailed enough and extensible enough to help designers find alternative and appropriate solutions.

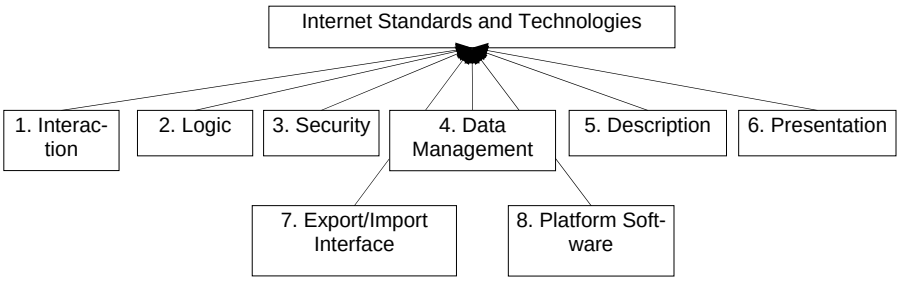

Fig. 4.1. Root classification entities

It is rather difficult to assign a technology or a standard to a single category bag. The reason for this is partly due to the fact that modern technologies and standards aim at integration and therefore cover multiple areas. A typical example for such a technology is ASP.NET – it represents a scripting technology and a Web invocation technology. Therefore the reader can expect that a single Web standard or technology is classified under multiple categories. Yet this needs to be minimized because it reduces the quality of the classification.

The fact that diverse technologies having some functional differences are classified under the same category is an issue. CB 3.1 (Fig. 4.5) is a typical example. Reflecting the primary objective for a technology is the guiding classification principle. Therefore it may happen that technologies having somewhat different orientation but still belonging to one group are put in one category bag.

Last but not least, finding the difference between a product, a technology, or a standard is a tricky issue. There are some technologies which are *de facto* standards (TCP/IP vs. ISO/OSI) and there are products which are based on a proprietary technology also having a rank of *de facto* standards (e.g. PGP). When performing classification or extending the one proposed it is important to concentrate on the general applicability and on the group to which the classified standard or technologies belongs. Do not discard products from further consideration when they represent a *de facto* standard, especially when classifying platform software (Sect. 4.1.8), but consider this choice carefully.

Categories	Example standards and technologies	Chapter
1. Interaction	ODBC, JDBC, SOAP, TCP/IP	2, 5, 7
2. Logic	Java Servlets, CORBA, EJB	5, 6
3. Security	XML Encrypt, LDAP	7, 10, 11
4. Data	HTML, XML, JAR	2, 6, 7, 8
5. Semantics	RDF, OWL	8
6. Presentation	HTML, XSL, CSS	5, 8
7. Export/Import Interface	WSDL, IDL	6, 7
8. Platform Software	Application and Web Server, database	2, 5, 6, 7, 8

Fig. 4.2. Discussion of the Internet standards and technologies

In this chapter we refer to many Internet standards and technologies. However, we avoid providing references to these since all of them are discussed in the second part of this book. Therefore, Fig. 4.2 indicates those chapters of this book where all the subsequently mentioned internet standards and technologies are explained. The reader is asked to refer to these chapters in order to get more information about them.

4.1.1 Interaction

The first group of technologies which will be considered is interaction (Fig. 4.3). In our view interaction is the application-specific part of the communication, i.e. the sequence of exchanged messages, the transmitted data structures, and so on and so forth. As mentioned in Sect. 2.5 it must distinguish between interaction and network communication. The latter is regarded as a capability of the platform.

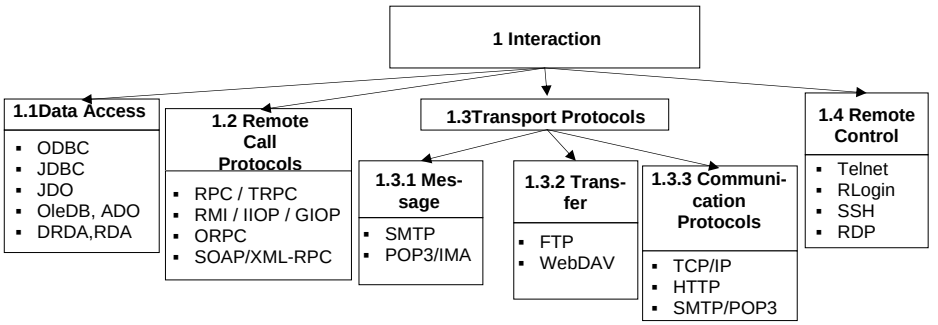

Fig. 4.3. Classification of interaction-related standards and technologies

We consider four subcategories, but are not limited just to them. For example, a new CB 1.5 could be added to account for the real-time protocols, if the designers decide to use any.

The first group of technologies is “Data Access” (CB 1.1). The primary focus of these technologies is to provide access to data collections of various types (predominantly databases). Despite the similarities, which is the reason for putting them in one category bag, there are also some genuine differences. ODBC and JDBC are competing technologies providing access to relational and object relational databases. OleDB and ADO (also ADO.NET) are database access technologies in the Microsoft realm based on top of ODBC. JDO (Java Data Objects) is a flexible object-oriented persistency mechanism. RDA (Remote Database Access, [ISO87]) and DRDA (Distributed Relational Database Architecture) [IBM04] [NeGr91] are technologies (standards) for remote database access. The support of database languages is not discussed explicitly. Most of the technologies are SQL based. Others, like JDO, support their own query language but are based on the ODMG standard.

The second category bag is CB 1.2 “Remote Call Protocols”. It contains a collection of protocols and technologies for remote procedure call-based invocations. These are mainly used in enterprise computing (middleware, component orientation). DCE RPC (Remote Procedure Call) is the most famous representative of this group. TRPC is a variant of RPC specially designed to provide transactional support to remote procedure calls.

IIOP and GIOP are CORBA-specific communication protocols. RMI (Remote Method Invocation) is Java-specific technology which is based on IIOP. ORPC is the equivalent technology for DCOM. SOAP (and its predecessor XML-RPC) is a lightweight protocol handling the communication in the realm of Web services.

The third category of interaction technologies is the abstract category 1.3 “Transport Protocols”. It is subdivided into three subcategories, namely message protocols, transfer protocols, and communication protocols. The category message protocols contain basically e-mail-related technologies such as SMTP (Simple Message Transport Protocol) used to send e-mail messages and POP3 (Post Office Protocol Version 3) and IMAP (Internet Message Access Protocol) to retrieve and organize e-mail messages. Interestingly enough, SMTP can also be used in the context of Web services as a communication protocol – therefore it also appears in CB 1.3.3. Protocols such as FTP (File Transfer Protocol) and WebDAV, which can be used to transfer files from one computer to another, are classified in CB 1.3.2 “Transfer Protocols”. There are probably many protocols which can be labeled as “communication protocols”, i.e. they can be used for application-specific interaction. Some of these protocols are TCP/IP, HTTP, the couple SMTP/POP3, and many others.

Last but not least, various protocols for remote control and operation can be classified under CB 1.4 “Remote Controls”. Technologies such as Telnet and SSH allow users to log on remotely, execute commands, and control the remote machine as if they were logged on locally. Telnet and Rlogin are nowadays succeeded by SSH (Secure SHell).

4.1.2 Logic

Logic (Fig. 4.4) forms the second category which will be considered. This category contains logic-related standards and technologies which are not directly related to the business application logic, which is the main reason for naming the category simply “Logic”. There are at least four subcategories: Scripting Languages, Business Logic, System Specific Logic, and Web Invocation Mechanisms.

CB 2.1 “Scripting Languages” is divided into two subcategory bags, CB 2.1.1 “Server Side” and CB 2.1.2 “Client Side”. CB 2.2 “Business Logic” classifies business logic-related standards into two category bags, CB 2.2.1 “Server Side” and CB 2.2.2 “Client Side”. As can be easily seen, the server side logic approaches are dominated by component-oriented technologies (COM/DCOM, EJB, etc.). The quite broad notion of J2EE has been written in the category bag. As discussed in Chap. 6 this implies that not only the EJB component technology but also the J2EE Web components can be used to implement a certain amount of business logic.

The next category bag is CB 2.3 “System Specific Logic“. The reason for calling this group system specific logic is that the application uses some of the capabilities provided by the system to program logic pieces. Database stored procedures are a very illustrative example. They represent a part of the application business logic associated with exclusively data-related operations programmed in a programming language supported by the database and stored and executed in it. If a data storage system different from a database were to be chosen then stored procedures would not have been available as a technological possibility and thus the designers would have had to think of an alternative solution. Further technologies are Web server or browser plug-in technologies or even executables or scripts implementing CGI, for example.

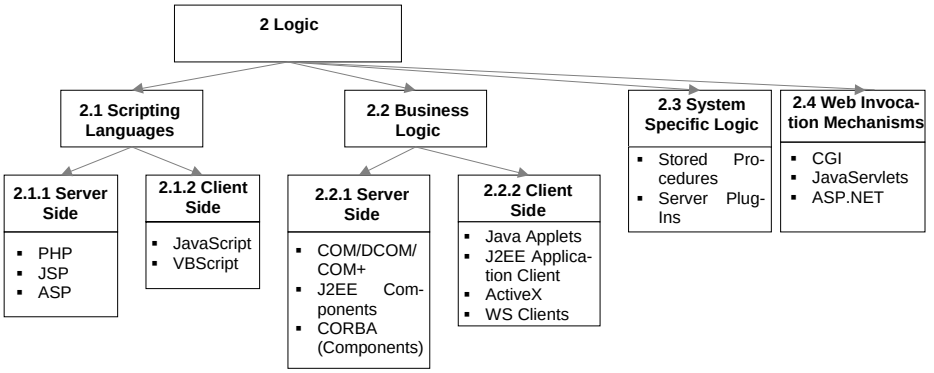

Fig. 4.4. Logic related standards and technologies

The last category bag CB 2.4 is "Web Invocation Mechanisms". It contains technologies such as CGI or Java servlets which are typically used to trigger the invocation of a method on business logic from an HTTP request. In the case of Java servlets this is not quite true. They can contain big and rather complex chunks of application logic. Still this does not diminish the fact that servlets can be used to trigger transactional and secure business logic method invocation.

4.1.3 Security

Security is the third classification category (Fig. 4.5). Although there are many alternative ways here it is preferred to classify the security-related standards and technologies into three different category bags: CB 3.1 "Message Security", CB 3.2 "Authentication", and CB 3.3 "Communication Security".

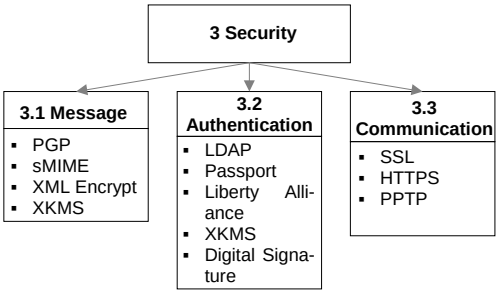

Fig. 4.5. Security-related Internet standards and technologies

Message security (CB 3.1) is a field of very active research, involving numerous competing technologies. As the name implies, the technologies gathered in this category serve the purpose of encoding just the contents of the exchanged documents or messages

in order to provide secure communication by leaving all other parties involved in the communication process unaffected. Some of these are PGP (Pretty Good Privacy), secure MIME (sMIME), XML Encrypt, and XKMS (XML Key Management Specification).

CB 3.2 contains technologies providing authentication services. These technologies are used as a means to identify and authenticate various users with the system and as a next step assign them the proper access control privileges and apply user preferences. Some of the technologies belonging to CB 3.2 are LDAP, XKMS, Microsoft Passport, and Liberty Alliance. Digital signatures are used to guarantee the authenticity of the digitally signed documents or messages, i.e. to make sure that the documents or messages indeed originate from the organizations or individuals claiming to be their authors.

Last but not least, several communication-level security-related technologies are classified under CB 3.3 “Communication Security”. These technologies are providing encrypted communication as a means for secure data transfer. In contrast to the technologies in CB 1.1, these provide encrypted communication channels leaving the messages unchanged. Such technologies are for example Secure Socket Layer (SSL), its derivative HTTPS, and PPTP (Point to Point Tunneling Protocol).

Web applications (except for intranet applications) are much more “exposed” to threats than regular applications running as part of an IT system within an enterprise. Apart from all the security technologies shortly presented here, there is a lot of work which has to be done by the platform software. For example, regardless of whether a Web application uses HTTPS to transfer critical data, system engineers need to make secure the configuration of the Web server. For example, the proper rights on the file system at OS level must be set. This process goes all the way down to the hardware architecture. For example, the routers must be properly configured.

4.1.4 Data

The fourth category (Fig. 4.6) contains the classification of some of the data-related standards and technologies. In this classification category we concentrate predominantly on files and file formats. Another major kind of data is the message format and the message data. It is left out here in favor of the interaction category where they actually need to be discussed. Another relevant issue is document versus message formats.

The reader will see a mixture of them in most of the category bags discussed in this section. XML is an interesting example in this respect – its primary goal is to be used as document format; there are, however, progressively more protocols formatting the messages in XML. To reduce complexity and increase readability the category bags will not be further subdivided in document and message categories.

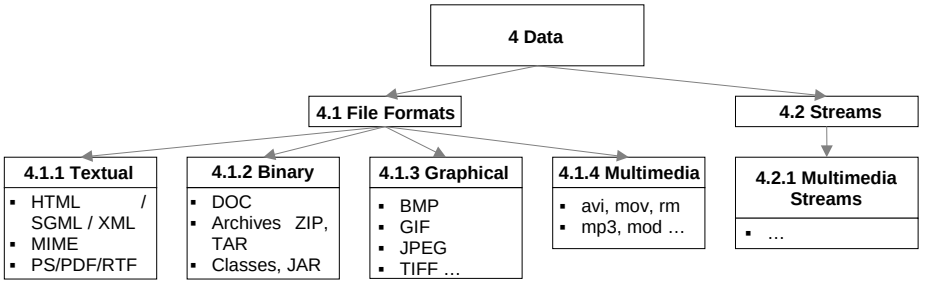

Fig. 4.6. Data-related standards and technologies

There are two subcategories of classification category “Data”, namely CB 4.1 “File Formats” and CB 4.2 “Streams”. CB 4.1 (File Formats) is subdivided into four category bags: CB 4.1.1 “Textual”, CB 4.1.2 “Binary”, CB 4.1.3 “Graphical”, and CB 4.1.4 “Multimedia”. There is such a wide variety of file formats on the Web that it is almost impossible to classify all of them in the proper category bag. Therefore it is attempted to classify just the most characteristic representatives in each category bag.

CB 4.1.1 contains standards and *de facto* file formats for textual documents and messages. File formats like HTML, SGML, or XML are markup-based document formats, which are also standardized. MIME (Multi-Purpose Internet Message Extension) is a standard widely utilized to format e-mail messages. Its use, however, is not limited just to e-mail messaging. Portable document formats are another issue to be reflected in the current category bag. File formats such as Adobe PDF (Portable Document Format), Adobe PostScript, and Microsoft RTF (Rich Text Format) are more or less *de facto* standards for documents created with only the goal of portability, i.e. operating system and device independence. For the most part these document formats are text based. There are, however, versions of these standards which are binary (e.g. linearized PDF).

CB 4.1.2 gives examples of binary file formats available on the Web. There are some examples of binary and proprietary document file formats such as Microsoft Office documents (based on the Compound Object Model) or Sun Open Office document formats. Additionally there are different archive formats such as ZIP, RAR, TAR, etc., which must be considered and also some executable files like for example Java archives and byte code .class files.

CB 4.1.3 contains some of the graphics formats available on the Internet. Formats such as GIF or JPEG or Bitmap or TIFF are standards on the Web.

Last but not least, some multimedia file formats are classified in this category. We chose not to develop the classification further and classify the standards into audio and audio/video. Certainly formats such as MPEG2, MPEG4, MPEG7, and file types such as AVI, MOV, or RM are part of this category bag. Some of the audio formats include .mp3, .mod, and many others.

4.1.5 Semantics

The semantics category contains standards for descriptive metadata (Fig. 4.7). It is subdivided into two subcategories: CB 5.1 “Web” and CB 5.2 “Multimedia”. The goal of using semantics-related technologies is to provide more and high-quality semantic descriptions, which can be used in searching and querying, composition, automated processing, automated reasoning, and many other fields. One of the major problems the Web faces today is the fact that there is quite a lot of information published. It is available in the form of structured or semi-structured documents. Unfortunately it is not “schematized” – that is, there are no schemata determining the type of a piece of information content. This is why the contents are mostly untyped – for example, the address of a person is simply available as text and is not of type address.

In the context of searching the missing schema leads to the fact that only text-based searches may be performed, i.e. string matching. This type of search leads normally to low-quality results. An attempt to solve this problem is made by introducing semantic descriptions. By doing so the contents are schematized and descriptive metadata attributes are assigned. Both of them are considered when searching, which improves significantly not only the search results but also the automated processing.

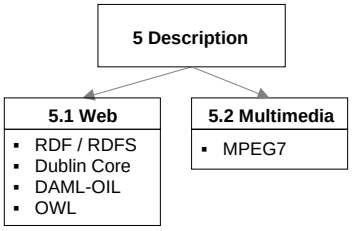

Fig. 4.7. Semantics-related Internet standards and technologies

CB 5.1 contains mainly ontology languages used to code semantic descriptions. Semantics and descriptive metadata play an important role not only on the Web but also in the field of multimedia where descriptions are becoming increasingly widespread with MPEG7. Such standards are used to describe structural parts of movie scenes such as characteristics of the persons on it or characteristics of the background. Organizing multimedia metadata requires extensive classification and standardization efforts.

4.1.6 Presentation

CB 6 contains standards and technologies used for presentation purposes (Fig. 4.8). In other words, these are technologies which are used to encode presentation data, which is then rendered (mostly graphically) by the client side platform.

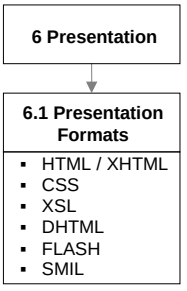

Fig. 4.8. Presentation-Related Internet standards and technologies

CSS (Cascading Style Sheets) guarantees a consistent layout of the presentation content regardless of the device or browser differences. Using XSL (XML Style Sheets) different transformations can be applied which eventually can be used to apply different presentation styles on the content. DHTML (Dynamic HTML) is a technology combining client side logic (VBScript or Java Script) with HTML. Flash and SMIL are proprietary multimedia extensions with which designers create high-quality animated multimedia Web sites.

4.1.7 Export/Import Interface

CB 7 contains standards for interface descriptions (Fig. 4.9). These can be used to define some parts of the application's business logic as the export interface. A description of the

interface made in one of these languages can be used for different purposes. The two most important are: to register the interface in a registry for discovery at later stage and to build subs and skeletons. The former determines, among other thing, why the term import interface is used. The application is actually imported by importing the construct derived from the interface description.

This idea is not genuinely new – the original term was API; then the idea gained significant importance with component-oriented programming. Web services are a technology which can be used to export the direct business logic interface in the context of Web applications.

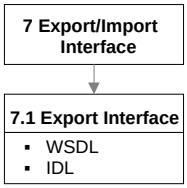

Fig. 4.9. Export/Import interface standards

IDL (Interface Definition Language) is a *de facto* standard for interface description originating from DCE RPC. It is used (with some changes and extensions) in CORBA and DCOM. WSDL (Web Service Description Language) is an interface description language for Web services, described in more detail in Chap. 7.

4.1.8 Platform Software

A classification of platform modules will be presented now (Fig. 4.10). Platform software is independent of the Web application architecture; however, it is preferred to discuss the platform software classification here because it logically belongs to the classification section. The different modules appearing in this classification will be used further in the platform design.

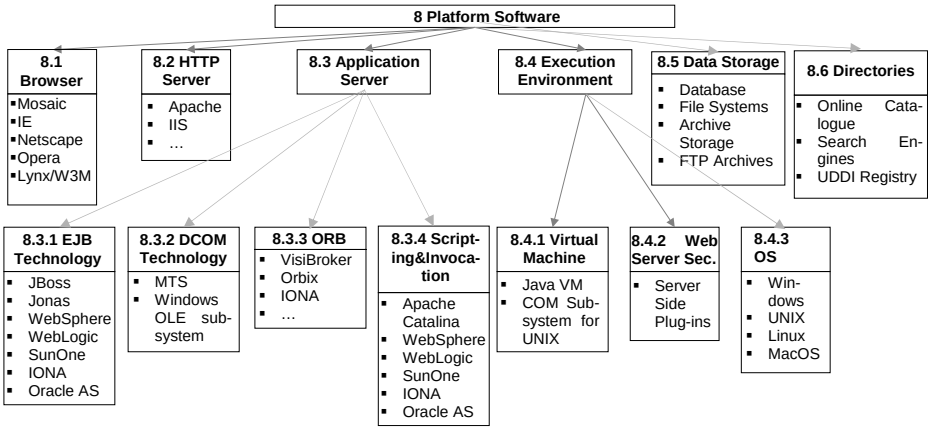

Fig. 4.10. Classification of some platform modules

There are several classification categories covering various platform-related technologies. This classification does not pretend to be exhaustive: there are platform modules which were intentionally left out; others require a higher level of detail.

Two interesting classification categories are CB 8.3 “Application Server and 8.4 “Execution Environment”. They are considered separately to underline the fact that there is a difference between execution environment for business logic components and execution environment for other WPA modules. Let us assume that large parts of the business logic of a Web application are written in EJB. The EJB components are deployed and executed in their container which is the EJB Server. The EJB server itself may be written in Java (e.g. JBoss) and must be executed as a “normal” Java application within the JVM. In this case the JVM is an execution environment for the EJB Server. We can apply the same reasoning with respect to the JVM and the OS. The JVM executes as a “normal” application on an OS and so on and so forth.

The application server category treats the notion of application server in the sense of component container. Three major branches can be distinguished: EJB component container, DCOM and COM+ component container, and CORBA ORB. Several implementations are available for these technologies some of which are listed in CB 8.3.1 through 8.3.3.

There are also several kinds of execution environments: virtual machines, Web servers, and operating systems (CB 8.4). Web servers are considered to act as the execution environment because some of them offer extensibility APIs for writing plug-ins. The server side plug-in executes in the environment of the Web server which controls their lifecycle, and offers memory management and resource control functions. The scientific community does not have a single position on whether or not this group must be classified under the application server group. Virtual machines such as the Java Virtual Machine are another kind of execution environment providing for portability.

4.2 Developing WAA and WPA – Continued

Having created a classification of technologies the designers are in a position to continue developing the architecture of the Web application in a stepwise manner. It was initiated in Sect. 2.2 and ended with the architecture of the Web application and the Web platform. At this stage the designers must make the decision of assigning technologies with which the architectural entities of the WAA will be implemented (Sect. 4.2.1). This decision can be based on the alternatives the classification offers and in a way reflecting as many as possible of the client requirements (Sect. 3.2); these requirements are meant when requirements are referenced throughout this section. The next step is to assign platform components to each module of the WPA (Sect. 4.2.2). To do so the designers consider the classification of platform modules.

4.2.1 Mapping the Technologies

Figure 4.11 is an extension of Fig. 3.11 and Fig. 3.12 showing the WAA components introduced in the matrix. To simplify the figure the columns in the matrix containing no entries were ignored. In order to put the technologies in the graphical representation, a gray box on top of each class or entity containing the technology in curly brackets is used.

The Generate button must be implemented using client side logic. JavaScript is a very good technological choice since it is natively supported by the large majority of Internet

browsers. Alternative technology from the same category bag is VBScript, however, it is browser and OS dependent, which contradicts requirement 1. Designers can also consider using a small Java applet as representative of CB 2.2.2. It is less likely to be chosen because it is not consistent with the thin-client ideology implied by requirement 2.

As discussed in Sect. 3.2, requirement 5, scripting must be used, which leads us to CB 2.1.1. On the other hand, we can assume that an implementation aligned with Java technologies is assumed due to the interoperability requirement. Therefore JSP can be used to implement `GenStartPage` and `GenOEPage` (Fig. 4.11). Since the only criterion was interoperability designers can also consider PHP, though a native Java technology implementation will lead to a coherent application. For similar reasons the choice of implementing the class `RequestHandler` in a Java servlet is made.

Requirement 5 calls for using a component technology to implement the major part of the business logic. The choice of EJB as the component technology to implement the prepare list is predefined. Last but not least, the choice of JDBC (CB 1.1, Fig. 4.3) appears logical as well.

4.2.2 Choosing Platform Software Modules

Having assigned different technologies the designers are now in a position to select platform software modules for each quadrant. To do so, designers may consider the classification of platform modules (Fig. 4.10).

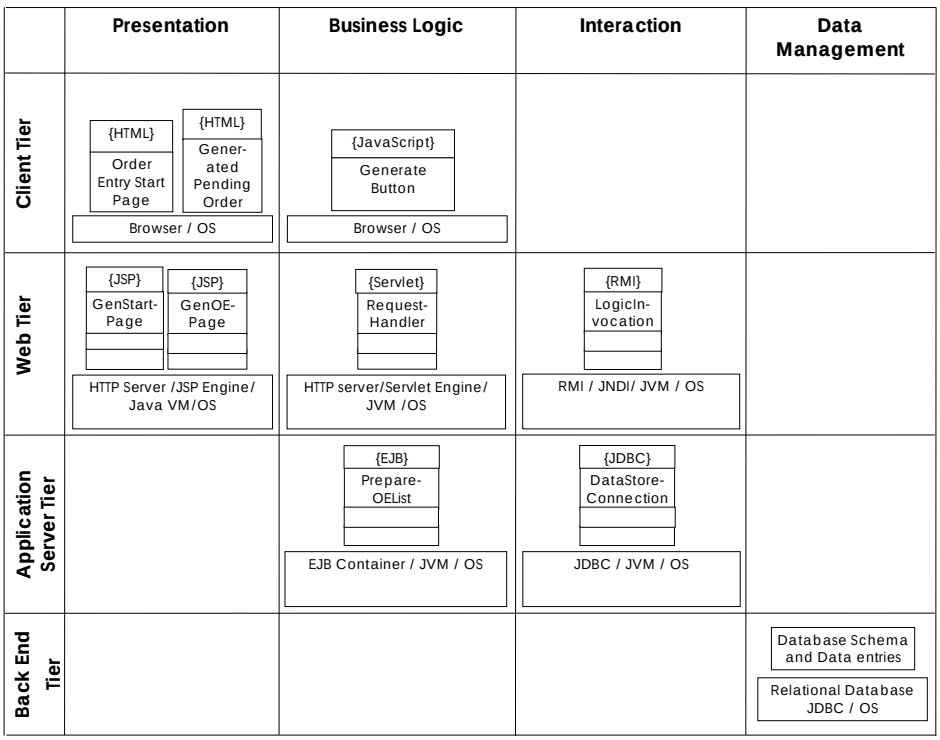

Fig. 4.11. Architecture of Web applications – continued

Any of the conventional Internet browsers (CB 8.1) will be in a position to effectively host all client tier entities. The thin-client architecture in the order entry example yields a minimal set of requirements. Therefore the client side platform is a simple one.

The platform of the Web requires an HTTP server by default. In order to account for the servlet and the JSP technologies (J2EE Web Components) the designers need to choose the appropriate engine. There is a single engine called J2EE Web components container for these two technologies (consider CB 8.3.4). All platform modules for Java technologies require the existence of a JVM (consider CB 8.4.1); therefore it must also be considered as platform software module. Additionally the designers need to choose an OS. The choice of an OS is arbitrary (no client requirement regarding the OS exists). Furthermore the JVM itself assures interoperability of the rest of the platform software.

No special platform software is required for RMI – all the needed technologies (JNDI, RMI) are included in the JVM.

An EJB container is necessary to implement the EJB business logic (consider CB 8.3.1). The designers need to make a strategic decision – whether to use an open source implementation or rather a commercial integrated application server product. For the purposes of the order entry application an open source implementation will deliver sufficient performance and reliability; therefore JBoss [JBOS04] or JONAS [JONA04] may be selected as EJB containers [SDK04].

Now that the matrix in Fig. 4.11 is constructed the designers are in a position to take a bird's eye view of the architecture and reevaluate it iteratively. By doing so the designers can reevaluate:

- The choice and the distribution of platform modules. Consider for example the Web tier – it is evident that almost all the required technologies are available in most of the software packages (commercial or open source) available today. Based on this architecture, however, the designers can pinpoint the precise use and discover and remove inconsistencies.
- The proper design of WAA packages and classes and their distribution over WPA tiers – the designers can review once again, considering the technology and platform software mapping and whether the chosen distribution is the proper one. It may well be the case that some new WAA classes come into play. Consider for example a future personalization, which will require storing the user preferences and settings on the client side platform. If they become too extensive, HTTP cookies will no longer be an appropriate solution. Hence WAA must be changed.
- New improvements – the quadrant <application server, presentation> is empty. A server side logging is precisely the right candidate to be positioned there. Having reached that conclusion the designers can implement a logger module to implement this functionality (which can later be used for auditing).

At this point we have reached a stage of the design which allows us to implement a first prototype or first operational version of the order entry application. As with all other design methodologies, our approach is also finally based on iterations and refinements.

Having defined the WAA and having chosen the WPA, developers can start generating tests for certain WAA components. If they are completely implemented and operational certain unit tests may be created automatically and synthetic data and function calls may be generated to empirically prove the proper functioning of certain WAA components.