2.1 Introduction

A thorough literature review is needed for the researcher to develop better understanding regarding the problem domain and the system to be developed. In this chapter, the researcher will explore deeper information about the current practices and information systems adopted by most of the companies involving in colour communication process of the textile industry, Web services technology, and the potential technologies that would be used to develop this project.

2.2 Colour Communication Process in Textile and Apparel Industry

During the decision making process of colour communication in textile industry, visual methods of colour matching can be haphazard and are notoriously unreliable for a number of reasons. These include the absence of standard illuminants and viewing conditions as well as the subjective and psychological nature of the human observer. There is often difficulty in agreeing standards with the added difficulty of establishing standard tolerances, while physical standards quickly become soiled. These shortcomings often lead to a significant quantity of wrong decisions throughout the colour communication process. (Park, 2007)

As early as 1970, McLaren (1970) quoted by Park (2007) had found that wrong decisions are likely to exceed 20% of assessments with trained colourists reversing their decisions on a similar proportion if given the opportunity to reassess colour standard and batch pairs. Park (2007) further argued that an improvement in the quality of colour matching, in terms of accuracy, consistency and reproducibility could only be achieved by the use of instrumental methods based on spectrophotometry and a colour-difference equation, which gave results that correlated with visual assessment by a panel of
experienced colour matchers. As a result, the so-called digital colour communication has become established as another means of achieving quick response in product development, including colour selection, together with the ability to monitor development and production without the need to ship physical colour samples.

Although digital colour communication technology has been widely applied in the field of colour communication in textile industry for years, most of the current colour communication systems adopted by the industry players are “semi-automated” system in which made up of separate pieces of platform dependent standalone applications that require a lot of additional human efforts.

At the starting point of textile and apparel industry colour supply chain, a colour specifier (apparel designer, colorant buyer) passes physical samples of target colour to different colour suppliers. Some of the colour suppliers with advanced colour measurement instrumentations will measure the reflectance values of different wavelength interval for each target colour by using spectrophotometer and directly save the digital colour readings into their respective standalone applications.

After that, the colour suppliers formulate several dying recipes that would produce lap dips (physical cloths that dyed to match a given colour standard) with colours that match the target colour. The colour suppliers then measure the reflectance values of each lap dip in order to obtain the digital reading of the matching colour. The digital reading of the lap dips will be used to be compared with the digital reading of the original colour specifications requested by the colour specifier.
Next, the colour suppliers send the digital comparison results between both of the target colour (original colour specification) and the match colour (measured from the physical lab dips) to the colour specifier via email or fax. At the same time, the colour suppliers may also send the physical samples of the matching colour (lap dips) to the colour specifier for visual reference by using courier service.

Finally, the colour specifier will decide whether the colour samples produced by a particular colour supplier fulfill the colour specification based on certain colour difference acceptance criteria. The mentioned acceptance criteria will take into the consideration the colour difference between the target colour and the match colour under primary illuminant as well as their colour differences under the viewing condition of other secondary illuminants to ensure the quality of colour constancy under different lighting conditions.

2.3 Digital Colour Communication Applications

In textile industry, digital colour matching technology has been well established for several decades for quality control of coloured materials against a standard. In addition to the contribution that digital colour matching technology makes to quick response and right-first-time processing, industry players also enjoy considerable financial savings by using non-physical colour matching standard based on reflectance measurements over the traditional physical standards which often get soiled quickly and hence tend to produce inaccurate colour matching results. (Park, 2007)

The latest advance in colour communication technology has utilized the power of Web-based solutions to address various diverse approaches to colour and to capture its complexities in a more comprehensive way.
In general, Web-based colour communication systems could reduce duplication of efforts, as well as streamline and enhance colour communication processes already in place by correcting the errors, omissions and deviations that happen throughout the colour supply chain at any time.

By using Web-based colour communication technology, textile manufacturers and their colour suppliers are able to communicate and assess colour visually and digitally. Rather than merely receiving spectral data in sets of numbers or solely receiving physical colour samples that lack of mathematical accuracy, the users of Web-based colour communication system are able to see precisely the colours on screen that correspond to a set of scientific colorimetric data. Thus, both of the visual tolerances and mathematical tolerances of the colour evaluation and comparison could be achieved realistically (Mulligan, 2002).

There are several vendors in market that provide application solutions to the colour communication process of the textile and apparel industry, such as Datacolor, eWarna and X-Rite.

### 2.3.1 Datacolor

Founded in 1970 and headquartered in Lawrenceville, New Jersey, United State of America, Datacolor is a global leader in colour management and colour communication technology. Datacolor initially developed solutions for industrial colour problems, and then expanded its product offering to include colour communication and digital imaging solutions. Industries supported by Datacolor include photography, digital imaging, graphic design, plastics, paint, leather, automotive, coatings, apparel, textiles, ink, printing, and paper. (Datacolor, 2007a)
Section 2.3.1.1 and section 2.3.1.2 discuss about colour communication applications provided by Datacolor.

2.3.1 Datacolor TRACK™

Datacolor TRACK™ is colour approval tracking tool in which enables the user to keep track the status and quality of colour throughout the supply chain. It is an integrated suite of software products, measuring systems, and support services that provides centralized control of the entire colour management process. It allows user to define views, filters and reports according to user’s preference. It records ‘status’ and ‘location’ information for each job in order to let everyone involves in colour communication process be informed about the upcoming required action items. (Datacolor, 2007b)

2.3.1.2 Datacolor MATCH TEXTILE™

Datacolor MATCH TEXTILE™ is a software application which could increase the accuracy of recipe calculation, and speed the colour matching process. It combines the power of intelligent software with an easy-to-use interface to provide powerful desktop tools, such as three-dimensional graphical displays and unique recipe archives to assist with dyestuff selection and accurate formulations. The intelligence in Datacolor MATCHTEXTILE™ would capture the past dyeing results and apply them in creating better formulation of recipes. The recipes can be converted to production dyelots with just a click of a button. (Datacolor, 2007c)
2.3.2 eWarna

Founded in 2000 and based in Malaysia, eWarna is a Web-based colour physics software company in which provides online colour management solution to the textile and apparel industry. It claims to be the first company which commercialized the world’s first global online colour management solution, Online Colour eXchange (OCX). OCX is the colour engine that allows users to store, retrieve, compare and share precise digital colour fingerprints anywhere, anytime. It provides programming interface for colour functions and uniform colour physics implementation across all applications in real time. Its principal products are LabWorks Pro, an award winning software application in which enable users to accurately measure, formulate, share, search and compare colour worldwide through an Internet enabled colour engine; and XMatch, a Web-based application for colour approval and workflow management process. (eWarna, 2004a; eWarna, 2004b; eWarna, 2005)

The following are colour communication applications provided by eWarna:-

2.3.2.1 LabWorks Pro

LabWorks Pro, an application built on the Online Colour eXchange engine, is a complete online colour management system containing the full range of standard colour analysis tools that enable users to accurately create, measure, formulate, share, search and compare colour worldwide through an Internet enabled colour engine. With LabWorks Pro, physical samples can be replaced with digital samples that can be authenticated and validated against the stored digital fingerprint of any colour. LabWorks Pro provides easy-to-use folder system for organizing colours, dyeloads, recipes and job formulations. The folder system provides the users with the facilities to copy and paste colour related entities, dyeloads, recipes and job formulations across different folders. Users are also allowed to share their respective folders with
colleagues, customers or suppliers in real time across the Internet. The copy and paste, and the sharing functions in the folder system enable user to reuse the proven standard colours, recipes, and job formulations during the Product Life Cycle thus saving the efforts and costs from creating needed colorimetric information from scratch. (eWarna, 2004a)

2.3.2.2 XMatch

XMatch is a Web-based application for colour approval and workflow management process in which can be accessed in anywhere around the world by using Web browser. In other words, by only using a Web browser to log on into XMatch account, retailers across the supply chain are able to communicate with vendors globally regarding colour quality control decisions in real-time. XMatch addresses various roles within the supply chain of textile industry such as buyer, specifier, middleman/agent, colour checker, vendor and mill. It allows users in different roles to communicate digitized colour standards at all times, receive and submit colour submissions (lab or bulk dips) through industry standard file formats, set approval tolerances and keep detailed records of all colour approvals. The application maintains an archived history of quality control decisions from which retailers can run real-time, global reports on lead time, length of colour approval process and quality of submissions. By providing retailers with a comprehensive view of vendor performance, XMatch enables them to leverage vendor relationships based on reliability, responsiveness and quality. (eWarna, 2004b)
2.3.3 X-Rite

Founded in 1958 and headquartered in Grand Rapids, Michigan, United State of America, X-Rite, Incorporated, develops colour management systems and solutions worldwide (BusnessWeek, 2009). X-Rite serves a wide range of industries, including printing, packaging, photography, graphic design, video, automotive, paints, plastics, textiles, dental and medical (X-Rite, 2007). Its products include colorimeters, spectrophotometers, spectrodensitometers, sensitometers as well as the software packages that allow the users to collect and store colour measurement data, compare data to databases, communicate colour results, and formulate colours from a database (BusnessWeek, 2009).

Below are colour communication applications provided by X-Rite:-

2.3.3.1 Color IMatch

Color IMatch textile colour formulation software is a software in which can be used to obtain accurate colour and effective formulations for textile colour recipe and enable electronic colour communication between retailers, specifiers and dyehouses throughout the world.

It could transform colour control from a series of unrelated colour samples and standards to a series of interrelated jobs. This workflow interface streamlines colour development by automating various matching parameters.

Using a networked database, formulas can be easily shared with other laboratories to ensure rapid colour development and consistency across all manufacturing facilities. The optional professional version of Color iMatch will include a new electronic sample simulator in which serves the purpose of simulating textured on-screen colour. Textured
on-screen colour provides more true-to-life sample simulation of textures such as knits and wovens. It accelerates communication of samples and lab dips between retailers and their suppliers throughout the world. (X-Rite, 2007)

2.3.3.2 Color iQC

Color iQC is a job oriented colour control software that takes the guesswork out of approving colours. It allows user to approve lab dips or production samples, manage multiple jobs concurrently, and inspecting finished goods. Operators are allowed to customize the job template according to their preference. The job template makes startup faster where the operator only needs to set up a job once and the software will store the display of job details in operator's preferred screen layout. (X-Rite, 2007)
### 2.3.4 Comparison Among Different Color Communication Applications in Textile and Apparel Industry

Table 2.1: Comparison Among Different Colour Communication Applications

<table>
<thead>
<tr>
<th>Vendor</th>
<th>DataColor</th>
<th>eWarna</th>
<th>X-Rite</th>
<th>Project to be developed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td>TRACK™ MATCH TEXTILE™ LabWorks Pro XMatch</td>
<td>Color iMatch Color iQC</td>
<td>Color Communication Web Service</td>
<td></td>
</tr>
<tr>
<td>Client/Server Architecture</td>
<td>Fat client architecture Stand alone program Fat client architecture Thin client architecture Fat client architecture</td>
<td>Fat client architecture Fat client architecture Fat client architecture</td>
<td>Fat server architecture</td>
<td></td>
</tr>
<tr>
<td>Functions</td>
<td>i. Colour approval tracking tool ii. Compare colour iii. Calculate and formulate dying recipe</td>
<td>i. Measure colour ii. Compare colour iii. Calculate and formulate dying recipe</td>
<td>i. Colour approval colour supply chain and workflow management tool ii. Compare colour iii. Calculate and formulate dying recipe</td>
<td>i. Colour approval and colour supply chain workflow management tool ii. Compare colour iii. Import colour from qtx file format iv. Export colours into qtx file format</td>
</tr>
<tr>
<td>Repository</td>
<td>Centralized Repository Repository attached to the program</td>
<td>Centralized Repository</td>
<td>Centralized Repository</td>
<td>Centralized Repository</td>
</tr>
<tr>
<td>Operation Setup</td>
<td>Client program installation Standalone program installation</td>
<td>Client program installation</td>
<td>No installation needed. Internet Browser-based Web application</td>
<td>Client program installation</td>
</tr>
<tr>
<td>Architecture design</td>
<td>Tightly coupling Tightly coupling</td>
<td>Tightly coupling</td>
<td>Tightly coupling</td>
<td>Tightly coupling Loosely coupling</td>
</tr>
</tbody>
</table>
Table 2.1 compares the differences between various current existing colour communication applications in the market with the proposed Web services.

From the aspect of client/server architecture, almost all existing applications supported by different colour communication vendors are falling in the category of fat client architecture. This is because these applications need to have programs with sophisticated processing logic installed on the client computer in order to perform complicated calculation. XMatch application is the only exceptional software as it is a Web-based workflow management tool which can be accessed by end users via normal Internet browser.

In terms of supported features, the functionalities supported by these softwares are varying depending on the market strategy adopted by each vendor. Software vendors tend to separate different functionalities into different pieces of software rather than include all features into one application so that they can market their products to different customer groups which each target customer group might only desire for few specific functionalities that are really relevant to their business operation nature.

In terms of database design, as colour communication softwares need to store data that is accessible to different stakeholders regardless of their geographical locations, software vendors have developed most of their applications on top of central repository except for the case of standalone application. For the only standalone application shown in table 2.1, Datacolor's MATCH TEXTILE and its database are residing in the same workstation.
From the point of view of architectural design, all of the existing vendor applications listed in Table 2.1 are tightly coupled softwares with low degree of software modularity, hence, this increases the difficulty of future software enhancement and reduces the possibility of software reusability.

As a result of the review on the current existing applications in the market, Table 2.1 also lists the general features that would be included in the design of the proposed Web services so that the development of colour communication Web services would adhere to the best practice of software engineering such as loosely-coupling and at the same time address the shortcoming of the existing vendor applications in the market.

The proposed Web services would be developed in a fat server architecture in order to minimize the processing burden residing at end user’s front-end client workstation. As the Web services need be accessed via SOAP messaging, end users would need to setup certain Web service API on their client workstations. Besides, the proposed Web services would be developed on top of a centralized repository so that the creation, modification and deletion of colour communication data could be done by all authorized Web services users regardless of their geographical locations.
2.4 Web Service

According to the definition by Kreger (2001) stated in the documentation of IBM Web Services Conceptual Architecture 1.0, a Web service is an interface that describes a collection of operations that are network accessible through standardized XML (Extensible Markup Language Protocol) messaging. A Web service is also defined as the software component that employed one or more of the following technologies in constructing a service which are Simple Object Access Protocol (SOAP), Web Services Definition Language (WSDL) or Universal Description, Discovery and Integration (UDDI) (Kumar & Subrahmanya, 2004).

2.4.1 Advantages of Web Services

According to Sienel et al (2007), Web services show the following advantages:

i. Web services provide interoperability between various software applications running on disparate platforms.

ii. Web services use open standards and protocols. Protocols and data formats are text-based where possible, making it easy for developers to comprehend.

iii. By utilizing HTTP, Web services can work through many common firewall security measures without requiring changes to the firewall filtering rules.

iv. Web services easily allow software and services from different companies and locations to be combined easily to provide an integrated service.

v. Web services allow the reuse of services and components within an infrastructure.
Interoperability is the key to business automation. It may be defined as the ability of an application interface and communicate with other applications on disparate systems of business partners, collaborators and make these available beyond geographical boundaries. Web service is characterized by its ability to stimulate the interoperability among communicating applications by using XML-based data representation. These applications are loosely coupled and could be synchronous or asynchronous in nature. (Kumar & Subrahmanya, 2004).

As the businesses grow and evolve, the need for the services continues to grow among all the related partners and collaborators. The need for business automation among these partners and collaborators is therefore on the raise. The technology of XML-based Web services appears to deliver the promise of interoperability among related business partners to bring about business automation (Kumar & Subrahmanya, 2004).

2.4.2 Web Service Architecture

A Web service is described by using a service description that adheres to the standard XML notion. The service description covers all of the necessary details that are needed to interact with the service such as message formats that detail the operations, transport protocols and location (Kreger, 2001).

The interface of the Web services hides the implementation details of each service, allowing the services to be used independently of the software platform and hardware which it is implemented. The interface also enables the services to be implemented independently of the programming language in which it is written. The platform-independent and programming-language-independent nature of Web services have allowed and encouraged Web Services-based applications to be loosely coupled,
component-oriented and interoperable over different technological implementations (Kreger, 2001).

The Web Services architecture is made up of the interactions between three roles, which are service provider, service registry, and service requestor. Service provider hosts a network-accessible software module that implements the Web services. It also defines a service description for the Web service and publishes it to the service requestor or service registry when it is being requested. Service requestor uses a find operation to directly retrieve the service description from the service provider or indirectly retrieve the service description from the service registry. Finally, the service requestor uses the service description to bind itself with the service provider followed by invoking the Web service implementation.

Figure 2.1 describes the interactions between the three prominent roles established in a Web service architecture: -

![Figure 2.1: Three prominent roles established in Web service architecture](Kreger, 2001)
Table 2.2 summarises the roles held by service provider, service requestor and service registry from both business perspective as well as architectural point of view.

Table 2.2: Description of prominent roles in Web service

<table>
<thead>
<tr>
<th>Role</th>
<th>Business Perspective</th>
<th>Architectural Perspective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service provider</td>
<td>It is the owner of the service.</td>
<td>It is the platform that hosts access to the service.</td>
</tr>
<tr>
<td>Service requestor</td>
<td>It is the business that requires certain functions to be satisfied.</td>
<td>It is the application that is looking for and invoking or initiating an interaction with a service. The service requestor role can be played by a browser driven by a person or a program without a user interface, for example another Web service.</td>
</tr>
<tr>
<td>Service registry</td>
<td>This is a searchable index that lists the information of all Web services registered with it.</td>
<td>It is a searchable registry of service descriptions where service providers publish their service descriptions. Service requestors find services and obtain binding information (in the service descriptions) for services during development for static binding or during execution for dynamic binding. However, for statically bound service requestors, it is an optional role in the architecture.</td>
</tr>
</tbody>
</table>
2.4.3 Web Services Related Technologies

2.4.3.1 Extensible Markup Language (XML)

Extensible Markup Language (XML) is a meta language to represent the data or a piece of information using markup (Kumar and Subrahmanya, 2004). XML was designed to be simple, extensible and to represent data in user defined way. In XML representation, information is being marked up with a tag. Any name that is systematically represented by a start angular brace < and an end angular brace > is called a tag, for example <studentName>. Information is represented with a pair of start tag and end tag, for example, <studentName>John</studentName>. The forward slash (/) is used on the tag name in the case of end tag.

World Wide Web Consortium (W3C) defines specifications that provide rules for representing XML data. Some of the most important rules are listed as below (Kumar and Subrahmanya, 2004): -

i. All XML elements must have a start tag and an end tag.
ii. All XML tags are case sensitive.
iii. All XML elements must be properly nested.
iv. All XML documents must have a root element

2.4.3.2 Simple Object Access Protocol (SOAP)

Simple Object Access Protocol (SOAP) which also known as "Service-Oriented Architecture Protocol" is a protocol specification that defines a uniform way of passing XML-encoded data. It also defines a way to perform Remote Procedure Calls (RPCs) by using HTTP as the underlying communication protocol (Vasudevan, 2001).
In 1997, Microsoft was started thinking about a XML-based distributed computing which could enable applications to communicate via Remote Procedure Calls by using a simple network of standard data types on top of XML/HTTP. During that time, two other companies, DevelopMentor and Userland were actively participating in the discussions (Graham et al, 2005).

Graham et al (2005) revealed that in September 1999, SOAP 0.9 was published for public review. In December of the same year, SOAP 1.0 specification was being officially released. The idea of SOAP was gaining a lot of momentum. In May 2000, SOAP 1.1 was submitted to the World Wide Web Consortium (W3C) as a note by IBM, Microsoft, UserLand, and DevelopMentor. In September 2000, the XML Protocols Working Group at W3C was formed to design the XML protocol that was to become the core of XML-based distributed computing. The group started with SOAP 1.1 as a foundation and produced SOAP 1.2 two years later, in June 2003.

In order to cater to the need of distributed information interchange, SOAP defines four conventions which are Message Envelope, a set of encoding rules, RPC convention, and binding with underlying transport protocol (Kumar & Subrahmany, 2004). Table 2.3 describes the roles played by each SOAP convention.
Table 2.3: Description of SOAP Conventions

<table>
<thead>
<tr>
<th>SOAP Conventions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message envelope</td>
<td>It defines a convention for describing the contents and packaging of the message.</td>
</tr>
<tr>
<td></td>
<td>It also defines how to route and process the message.</td>
</tr>
<tr>
<td>Encoding rules</td>
<td>These rules provide convention of mapping the application data to XML based data.</td>
</tr>
<tr>
<td></td>
<td>These rules are based on XML Schema and as such closely model many of the standard data types and constructs that most of the programming languages provide.</td>
</tr>
<tr>
<td>RPC convention</td>
<td>SOAP specification defines rules for carrying out Remote Procedure Calls (RPCs). RPC calls request-response oriented calls that take care of how data is sent through request call and received the result through the response.</td>
</tr>
<tr>
<td>Binding with underlying transport protocol</td>
<td>Although SOAP can be exchanged over many transport protocols, the only binding defined in the SOAP specification is to HTTP.</td>
</tr>
</tbody>
</table>

(Graham et al, 2005)

2.4.3.3 XML Messaging Using SOAP

In XML messaging-based distributed computing, the basic requirements for a network node to play the role as Web service requestor or Web service provider are the ability to produce, parse a SOAP message, or both, and the ability to communicate over a network (Kreger, 2001).

Basically, a SOAP server that runs on a Web application server performs the functions mentioned in the previous paragraph. Alternatively, a programming language-specific runtime library can be used to encapsulate these functions within an Application Programming Interface (API).
In IBM’s Web Services Conceptual Architecture 1.0 white paper, Kreger (2001) elaborated the basic steps of the workflow for an application that is integrated with SOAP as below:-

![Diagram of XML messaging using SOAP](image)

**Figure 2.2: XML messaging using SOAP**
(Kreger, 2001)

i. In Figure 2.2, at (1) A Web service requestor’s application creates a SOAP message. This SOAP message is the request that invokes the Web service operation provided by the service provider. The XML document in the body of the message can be a SOAP Remote Procedure Call (RPC) request or a document-centric message as indicated in the service description. The service requestor presents this message together with the network address of the service provider to the SOAP infrastructure. The SOAP client runtime interacts with an underlying network protocol to send the SOAP message out over the network.

ii. At (2) the network infrastructure delivers the request message to the service provider’s SOAP runtime (for example a SOAP server). The SOAP server routes the request message to the service provider's Web service. The SOAP runtime is responsible for converting the XML message into programming language-
specific objects if required by the application. This conversion is governed by the encoding schemes found within the message.

iii. The Web service is responsible for processing the request message and formulating a response. The response is also a SOAP message. At (3) the response SOAP message is presented to the SOAP runtime with the service requestor as the destination. In the case of synchronous request/response over HTTP, the underlying request/response nature of the networking protocol is used to implement the request/response nature of the messaging. The SOAP runtime sends the SOAP message response to the service requestor over the network.

iv. At (4) the response message is received by the networking infrastructure on the service requestor’s node. The message is routed through the SOAP infrastructure; potentially converting the XML message into objects in a target programming language. The response message is then presented to the application.
2.4.3.4 Web Services Description Language (WSDL)

Web Service Definition Language (WSDL) can be seen as a complement to SOAP, as it facilitates interoperability between Web services. It is used to describe the message syntax associated with the invocation and the response of a Web service. The specifications for WSDL were come from a joint initiative by Microsoft, IBM and Ariba (Lefebvre, 2002). WSDL version 1.1 was submitted to World Wide Web Consortium (W3C) for standardization in March of 2001. WSDL 1.1 forms the basis of the current standard version, WSDL 2.0 (Graham et al, 2005).

Like Interface Definition Language (IDL), which acts as a service describer with Common Object Request Broker Architecture (CORBA), WSDL is a XML syntax to describe Web services. More and more SOAP implementations support this description language. With WSDL, applications that use SOAP can self-configure exchanges between Web services, while hiding most of the low-level technical details (Lefebvre, 2002).

As described by Table 2.4 on the following page, WSDL used by the service provider to stores the fundamental properties of a Web service. It provides a way for service providers to describe the basic format of web service requests over different protocols or encodings. It describes what a web service does, where a service is located, and how a service is accessed (Graham et al, 2005; Vasudevan, 2001).
### Table 2.4: Fundamental Web Service properties described by WSDL

<table>
<thead>
<tr>
<th>Fundamental Web Service properties described by WSDL</th>
<th>Elaboration</th>
</tr>
</thead>
<tbody>
<tr>
<td>What a web service does?</td>
<td>The operations (methods) the service provides, and the data (arguments and returns) needed to invoke them.</td>
</tr>
<tr>
<td>Where a service is located?</td>
<td>Details of the protocol-specific network address such as Universal Resource Locator (URL).</td>
</tr>
<tr>
<td>How a service is accessed?</td>
<td>Details of the data formats and protocols necessary to access the service’s operations.</td>
</tr>
</tbody>
</table>

With WSDL, the service provider can easily communicate all the specifications for invoking the Web service to the service requestor through the service description. The service description combined with the underlying SOAP infrastructure encapsulates implementation details away from the service requestor's application and the service provider’s Web service. As such, it enables the Web Services architecture to be loosely coupled and at the same time reduces the amount of required shared understanding over the custom programming and integration between the service provider and the service requestor (Kreger, 2001).

#### 2.4.3.5 Universal Description, Discovery and Integration Service (UDDI)

Universal Description, Discovery and Integration Service (UDDI) provides a mechanism for clients to dynamically find other Web services. Via the usage of UDDI interface, businesses can be dynamically connected to services provided by external business partners (Vasudevan, 2001).

A UDDI registry is similar to a CORBA trader or Dynamic Naming Server (DNS) service for business applications. A UDDI registry has two kinds of clients which are businesses that want to publish Web services via their usage interfaces, and clients who want to obtain certain types of services and bind programmatically to them (Vasudevan, 2001).
UDDI facilitates service discovery at both design time and runtime. In typical Web services scenarios, service providers want to publish their service descriptions to a registry, and service requestors want to query service registry for service description during either design time or runtime (Graham et al, 2005).

The UDDI initiative was first announced on September 2000. The first three UDDI versions were developed by UDDI.org as the sponsoring organization. UDDI.org submitted UDDI 3.0 to the Organization for the Advancement of Structured Information Standards (OASIS) so that the specifications can be evolved into formal standard. In April 2003, UDDI 2.0 specifications were approved as formal OASIS standards. Currently, UDDI 3.0 specifications have been published as OASIS committee specifications (Graham et al, 2005).

2.4.4 Web Service Stack

To perform the three operations of publish, find and bind in an interoperable manner, there must be a Web Services stack that embraces standards at each level. Figure 2.3 illustrates a conceptual Web Services stack where the capabilities of upper layers are built upon the capabilities provided by the lower layers.

![Figure 2.3: The Conceptual Web services stack (Kreger, 2001)](image-url)
The bottom layers of this stack, representing the base Web services stack, are relatively mature and more standardized than the layers higher in the stack. The vertical towers represent requirements that must be addressed at every level of the stack. The text on the left represents standard technologies that apply at that layer of the stack. The maturation and adoption of Web services will drive the development and standardization of the higher levels of the stack and the vertical towers. (Kreger, 2001)

Kreger (2001) summarizes the functions of each conceptual Web Services stack as follows:-

i. Network
The foundation of the Web services stack is the network. Web services must be network accessible in order to be invoked by a service requestor. Web services that are publicly available on the Internet use commonly deployed network protocols. Because of its ubiquity, Hypertext Transfer Protocol (HTTP) is the de facto standard network protocol for Internet-available Web services. Other Internet protocols can be supported, including Simple Mail Transfer Protocol (SMTP) and File Transfer Protocol (FTP).

ii. XML-based messaging
The next layer, XML-based messaging, represents the use of XML as the basis for the messaging protocol. SOAP is the chosen XML messaging protocol for the following reasons:

- It is the standardized enveloping mechanism for communicating document-centric messages and remote procedure calls using XML.
- It is simple as it is basically an HTTP POST with an XML envelope as payload.
- It is preferred over simple HTTP POST of XML because it defines a standard mechanism to incorporate orthogonal extensions to the message using SOAP headers and a standard encoding of operation or function.
- SOAP messages support the publish, find and bind operations in the Web services architecture.

iii. Service Description

The service description layer is actually a stack of description documents. First, Web Services Description Language (WSDL) is the de facto standard for XML-based service description. This is the minimum standard service description necessary to support interoperable Web services.

WSDL defines the interface and mechanics of service interaction. Additional description is necessary to specify the business context, qualities of service and service-to-service relationships. The WSDL document can be complemented by other service description documents to describe these higher level aspects of the Web service. For example, business context is described using Universal Description, Discovery and Integration Service (UDDI) data structures in addition to the WSDL document. Service composition and flow are described in a Web Services Flow Language (WSFL) document.

Because a Web service is defined as being network-accessible via SOAP and represented by a service description, the first three layers of this stack are required to provide or use any Web service. The simplest stack would consist of HTTP for the network layer, the SOAP protocol for the XML messaging layer and WSDL for the service description layer. This is the interoperable base stack that all inter-enterprise, or public Web services should support. Web services, especially intra-enterprise, or private
Web services, can support other network protocols and distributed computing technologies. Figure 2.4 depicts the interoperable base stack.

![Interoperable base Web services stack](image)

Figure 2.4: Interoperable base Web services stack

(Kreger, 2001)

The stack depicted in the Figure 2.4 provides for interoperability and enables Web Services to leverage the existing Internet infrastructure. This creates a low cost of entry to a ubiquitous environment.

iv. Service Publication

Any action that makes a Web Services Description Language (WSDL) document available to a service requestor, at any stage of the service requestor’s lifecycle, qualifies as service publication. The simplest example at this layer is the service provider sending a WSDL document directly to a service requestor. This is called direct publication. E-mail is one vehicle for direct publication. Direct publication is useful for statically bound applications. Alternatively, the service provider can publish the WSDL document describing the service to a host local WSDL registry, private Universal Description, Discovery and Integration Service (UDDI) registry or the UDDI operator node.
v. Service Discovery

Because a Web service cannot be discovered if it has not been published, service discovery depends upon service publication. The variety of discovery mechanisms at this layer parallels the set of publication mechanisms. Any mechanism that allows the service requestor to gain access to the service description and make it available to the application at runtime qualifies as service discovery. The simplest example of discovery is static discovery wherein the service requestor retrieves a Web Services Description Language (WSDL) document from a local file. This is usually the WSDL document obtained through a direct publish or the results of a previous find operation.

Alternatively, the service can be discovered at design time or runtime using a local WSDL registry, a private Universal Description, Discovery and Integration Service (UDDI) registry or the UDDI operator node.

vi. Service Flow

Because a Web service’s implementation is a software module, it is natural to produce Web services by composing Web services. A composition of Web services could play one of several roles. Intra-enterprise Web Services might collaborate to present a single Web service interface to the public, or the Web services from different enterprises might collaborate to perform machine-to-machine, business-to-business transactions. Alternatively, a workflow manager might call each Web service as it participates in a business process. The topmost layer, service flow, describes how service-to-service communications, collaborations, and flows are performed. Web Services Flow Language (WSFL) is used to describe these interactions.
vii. Vertical Towers

For a Web services application to meet the stringent demands of today’s e-businesses, enterprise-class infrastructure must be supplied, including security, management and quality of service. These vertical towers must be addressed at each layer of the stack. The solutions at each layer can be independent of each other. More of these vertical towers will emerge as the Web services paradigm is adopted and evolved.

2.4.5 Web Services in Market

2.4.5.1 Amazon Web Services

Before year 2006, Amazon provided its Web services via its Amazon Affiliate Program which is now known as Amazon Associate Program. Via the Amazon Web services developed for Amazon Affiliate Program, Amazon allows remote access to its inventory database, by providing free Amazon Web services developer kits that enable others to produce faster searches of "light" versions of the company's catalogue (Kane, 2002).

Amazon’s Web services for Amazon Associate Program are self-contained functions that can be published and invoked across the Internet by using XML-based protocols. These Web services provide functions for directly accessing Amazon's technology platform and product data, ranging from retrieving information about target products to adding desired items to shopping cart. Third party Website developers and Website administrator can access Amazon Web Services via an XML over HTTP (also called "XML/HTTP" or “REST”) or a SOAP interface (BusinessWeek, 2010).

Amazon Web services for its affiliate program had been designed in such a way that they can be used by users in few steps. Basically what a Web services developer needs to do can be summarized in three steps which are download the developer's kit from Amazon Web services official Web site, obtain a developer's token from the
authentication email after completing the sign up Web form, and finally write application that could utilize Amazon Web services (DuCharme, 2004).

To date, other than providing Web services that expose Amazon's product data and e-commerce functionalities to the third party applications, Amazon also delivers is Web services via cloud computing technology. These cloud-based Web services include:-

i. Amazon Elastic Compute Cloud - Web services that provide resizable compute capacity in the cloud.

ii. Amazon Flexible Payments Service - Web services which allow the movement of money between any two entities, humans, or computers.

iii. Amazon Mechanical Turk- Web services that provide API for computers to integrate artificial intelligence directly into their processing.

iv. Amazon Simple Storage Service - Web services that provide interface to store and retrieve data.

v. Amazon Relational Database Service - Web services that allow user to set up, operate, and scale relational databases in the cloud.

vi. Amazon Simple Queue Service - Web services that offer a hosted queue for storing messages as they travel between computers.
2.4.5.2 Google Web Services

According to Kane (2002), since 2002, Google has been experimenting an application of Web services that gives developers direct access to its search database, bypassing its Website and allowing them to design their own ways to use the provided search engine technologies.

Kane (2002) further pointed out that Website developers or Website administrators can develop their applications that integrate with Google Web services by making use of Google Web APIs (application programming interfaces). These custom APIs allow developers to tailor the content of Google search engine database based on their respective specific needs. With the Google Web APIs service, software developers can query billions of Web pages indexed by Google database directly from their own application programs.

Developers write software programs that connect remotely to the Google Web APIs service via the Simple Object Access Protocol (SOAP), an XML-based mechanism for exchanging typed information. The platform independent SOAP and WSDL standards enable developers to make use of Google Web services on their favourite development technology environment such as Java, Perl, or Visual Studio .NET.

The Google Web APIs service introductory page describes the three steps of using Google Web services. According to the introductory page, the developers just need to download the developer’s kit, followed by creating a Google account, and finally write their programs by using authorized license key that enables the programs submit valid queries to the Google Web APIs service (Google, 2006a). Other than issuing search requests to Google's index of Web pages and receive results as structured data, the developers can also access information in the Google cache, and check the spelling of
words. Google Web APIs enable the applications developed by the third party individuals support the same search syntax as the Google.com site (Google, 2006b).

The applications that developers can create by using Google Web APIs include (Google, 2006b):

i. Issuing regularly scheduled search requests to monitor the Web for new information on a subject.

ii. Performing market research by analyzing differences in the amount of information available on different subjects over time.

iii. Searching via non-HTML interfaces, such as the command line, pagers, or visualization applications.

iv. Creating innovative games that play with information on the Web.

However, Google has formally phased out Google SOAP Search API starting from 7th September 2009 (Google, 2009a).

Now, Google has shifted its technological approach of offering its variety of online services via Google AJAX Search API. Instead of applying open Web services standards, such as WSDL, SOAP and UDDI, Google AJAX Search adopts JavaScript as the API programming medium to let third party application developers embed Google Search services in their Web pages (Google, 2009b). The Google search services offered via Google AJAX Search API include Web Search (Web Search, News Search, and Blog Search), Multimedia Search (youtube video search and image search) and Local Search (Google Map Search).
2.5  Technologies Used for Project Development

2.5.1  Java Programming Language

The Java programming language is a high-level language that can be characterized as object-oriented, distributed, interpreted, portable, robust and multithreaded.

The Java programming language is designed to be object-oriented from the ground up. Programmers are able to access existing libraries of tested objects that provide functionality ranging from basic data types through I/O and network interface to graphical user interface toolkits. These libraries can be extended to provide new behaviour. (Sun Microsystems, 1997)

With most programming languages, the program must be either compiled or interpreted so that it can be executed on a machine. However, Java programming is different from the others in which that program written in Java is both compiled and interpreted. First, the java program, will be translated into Java bytecodes. Then, the platform-independent Java bytecodes will be interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java bytecode instruction on the computer. Compilation happens just once; while interpretation occurs each time when it is executed. (Sun Microsystems, 1997)

A Java program can be compiled into bytecodes on any platform that has a Java compiler. The bytecodes can be run on any interpretation of the Java Virtual Machine (JVM). As long as a computer has a Java Virtual Machine installed on it, the same program written in Java can be run on it regardless what operating system the computer is based on. (Sun Microsystems, 1997)
2.5.2 MySQL

In 1994, TcX, a Swedish consulting firm, needed a fast and flexible way to access their tables. Unable to find a database server that could accomplish the required task adequately, Michael Widenius, the principal developer at TcX, decided to create his own database server. The resulting product was called MySQL (Deitel et al., 2002).

MySQL is a multi-user, multithread relational database server (Deitel et al., 2002; Lindgren, 1999). It can run on most unix platforms, windows and OS/2. Relational databases store information in tables with rows and columns. It uses SQL (Structured Query Language) to interact with and manipulate data (Deitel et al., 2002). SQL is high-level, declarative database language used for defining and querying relational databases (Lindgren, 1999).

MySQL shows the following features (Deitel et al., 2002):

i. Multithreading capabilities that enable the database to perform multiple tasks concurrently, allowing the server to process client requests efficiently.

ii. Support for various programming languages such as C, C++, Java, Python, PHP, Perl and etc.

iii. Implementations of MySQL are available for Windows, Linux and Unix.

iv. The ability to access tables from different databases by using a single query, thus, increasing the efficiency of retrieving accurate and necessary information.

v. The ability to handle large databases.

MySQL is becoming the database of choice for many businesses, universities, and individuals. MySQL’s rising popularity benefits from the open source software movement (Deitel et al., 2002).
2.5.3 Apache Tomcat Server

The Apache Tomcat server is an open source, Java-based Web application container that was created to run Servlets and JavaServer Pages (JSP) in Web applications. Tomcat is supported and maintained under the Apache-Jakarta subproject by volunteers from the open source Java community (Osborne, 2002). As part of Apache's open source Jakarta project, it has nearly become the industry accepted standard reference implementation for both the Servlets and JSP API (Goodwill, 2002). Apache SOAP service can be integrated into Apache Tomcat server in order to build up Web service.

2.5.4 Apache AXIS

AXIS stands for Apache eXtensible Interaction System. According to Apache AXIS Website’s introductory page (Apache Axis, 2004), Axis is an implementation of the SOAP (Simple Object Access Protocol) submission to the World Wide Web Consortium (W3C), the main international organization that creates open standards for the World Wide Web.

According to Graham et al (2005), AXIS has its origin as a SOAP implementation SOAP4J project in IBM. In 1999 IBM contributed an early implementation of the SOAP protocol to Apache Foundation, which became known as Apache SOAP. However, this implementation which based on the earlier work call SOAP4J, was written in a monolithic style. It was lacking of modularity and was very tight coupling (Graham et al, 2005).

In 2000, the Apache SOAP community decided to major re-architect the implementation framework to increase its degree of extensibility and as well as improve its performance. As such, around the time when Apache SOAP version 2.1 has formally released, the development team was focusing on the effort in soliciting input on Apache
SOAP 3.0 which would be seeing a major re-factoring of the Apache SOAP code base (Graham et al, 2005).

As mentioned in the opening of this section, AXIS is the short for Apache eXtensible Interaction System. This name was chosen instead of Apache SOAP 3.0 (the original plan) because at that time, the XML Protocol working group at W3C was just getting underway, and they believed the protocol was going to be called XP rather than SOAP – as such, having SOAP in the name seemed retro (Graham et al, 2005).

As the implementation of SOAP 3.0 finally turned out, the development community decided to name it as Apache eXtensible Interaction System to signal the extensibility that was achieved with the new architecture of the implementation framework as well as to differentiate it from other SOAP implementations (Graham et al, 2005).

The main architectural idea behind AXIS is that of chains of message-processing components, can be developed separately and assembled at deployment time. These components called handlers, can each process portions of the message or do other custom work and can be combined to generate powerful and flexible systems (Graham et al, 2005).

According to the Apache Web Services Wiki (2006), AXIS version 1.x supports the following open standards:-
- W3C's SOAP version 1.1 and SOAP version 1.2
- W3C's WSDL version 1.1.
- Sun Microsystem's SAAJ (Sun SOAP with Attachments API for Java) version 1.1
- Sun Microsystem's Java API for XML-Based RPC (JAX-RPC) version 1.0.
2.6 Colour Physics

2.6.1 Visible Light

In the book “Colour in Computer Graphics”, Lilley et al (1993) pointed out that light is a form of electromagnetic energy which could be considered to behave like a wave, and the factor that distinguishes between visible light and many other types of electromagnetic energy is the wavelength. Visible wavelengths are most conveniently measured in nanometres (nm, \(10^{-9}\) m).

![Wavelength Chart](image)

Figure 2.5: The wavelength range of visible light in electromagnetic spectrum (Lilley et al, 1993)

The ranges of wavelengths which broadly correspond to the colours of the spectrum are shown in Table 2.5 on the next page.
Table 2.5: Approximate wavelengths of spectral colours

<table>
<thead>
<tr>
<th>Range (nm)</th>
<th>Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>380 – 450</td>
<td>Violet</td>
</tr>
<tr>
<td>450 – 490</td>
<td>Blue</td>
</tr>
<tr>
<td>490 – 560</td>
<td>Green</td>
</tr>
<tr>
<td>560 – 590</td>
<td>Yellow</td>
</tr>
<tr>
<td>590 – 640</td>
<td>Orange</td>
</tr>
<tr>
<td>640 – 730</td>
<td>Red</td>
</tr>
</tbody>
</table>

(Lilley et al, 1993)

2.6.2 Illuminants

Lilley et al (1993) explained that objects are coloured which do not themselves emit light but the visual effect of colour is due to the reflection of light from other light sources. Objects absorb varying amounts of light of each wavelength, and the unabsorbed portion is reflected back to the eye to give the sensation of colour. Therefore, the quality of light illuminating the object will affect the perceived colour.

After cross-referencing from multiple sources which specifying CIE standard illuminant data, DiCosola (1995) summarized the main illuminants widely used by various industries as follows:-

i. Illuminant A - Incandescent

The CIE standard illuminant A was first recommended in 1931 to represent incandescent light sources with an approximate colour temperature of 2856 degrees Kelvin.
ii. Illuminant C - Daylight

CIE standard illuminant C was recommended in 1931 to represent average daylight with a colour temperature of 6774 degrees Kelvin.

iii. Illuminant D50 - Daylight @ 5000K

CIE standard illuminant D50 is correlated colour temperature D illuminant which is calculated from the standard illuminant D65. This standard illuminant represents Daylight at 5000 degrees Kelvin and is the ANSI standard illuminant used in the graphic arts industry.

iv. Illuminant D65

The CIE standard illuminant D65 represents daylight at approximately 6500 degrees Kelvin. This standard and the method for calculating different correlating colour temperatures was introduced in 1964.

v. Illuminant F2 - Cool White

The CIE standard illuminant F2 represents the typical cool white fluorescent source.

vi. Illuminant F7 - Broad Band Daylight

The CIE standard Illuminant F7 represents a broad band daylight fluorescent source. Examples of this type of source are the Daylight fluorescent sources manufactured by GE and Philips.

vii. Illuminant F11 - Narrow Band White

The CIE standard illuminant F11 represents a narrow band white fluorescent source. Sources similar to this illuminant are the Philips F40AX41 and TL841, as well as the GE SPX41.
2.6.3 The CIE standard observer

In the study of the perception of colour, one of the first mathematically defined colour spaces was the CIE 1931 XYZ colour space, created by the International Commission on Illumination (CIE) in 1931. (Wikipedia, 2005a)

The CIE XYZ colour space was derived from a series of experiments done in the late 1920s by W. David Wright and John Guild. Their experimental results were combined into the specification of the CIE RGB colour space, from which the CIE XYZ colour space was derived.

In the CIE XYZ colour space, the tristimulus values are a set of tristimulus values called X, Y, and Z, which are roughly red, green and blue, respectively. Two light sources, made up of different mixtures of various wavelengths, may appear to be the same colour; this effect is called metamerism. Two light sources have the same apparent colour to an observer when they have the same tristimulus values, no matter what spectral distributions of light were used to produce them.

Due to the nature of the distribution of cones in the eye, the tristimulus values depend on the observer's field of view. To eliminate this variable, the CIE defined the standard (colorimetric) observer.

Originally this was taken to be the chromatic response of the average human viewing through a 2° angle, due to the belief that the color-sensitive cones resided within a 2° arc of the fovea. Thus the CIE 1931 Standard Observer is also known as the CIE 1931 2° Standard Observer.
Because the CIE 1931 2° Standard Observer is not really appropriate for large-field visual colour judgements, the CIE defined a second set of observer functions in 1964 known as the supplementary observer data based upon colour-matching experiments with a field of 10 degrees. Since the 2 degree data are still in use, the CIE 1964 10° Standard Observer data are often differentiated from the original 1931 data by the use of subscripts.

2.6.4 Colour Measurement and Representation

Colour appearance of a given object can be measured into digital mathematical value by using reflectance spectrophotometer. Reflectance spectrophotometers measure the amount of light reflected by a surface as a function of wavelength to produce a reflectance spectrum. The reflectance spectrum of a sample can be used, in conjunction with the CIE standard observer function and the relative spectral energy distribution of an illuminant, to calculate the CIE XYZ tristimulus values for that sample under that illuminant. (Colourware, 2009)

The operation of a spectrophotometer is basically to illuminate the sample with white light and to calculate the amount of light that is reflected by the sample at each wavelength interval. Typically data are measured for 31 wavelength intervals centred at 400nm, 410nm, 420nm until 700nm. This is done by passing the reflected light though a monochromating device that splits the light up into separate wavelength intervals. The instrument is calibrated using a white tile whose reflectance at each wavelength is known compared to a perfect diffuse reflecting surface. The reflectance of a sample is expressed between 0 and 1 (as a fraction) or between 0 and 100 (as a percentage). It is important to realize that the reflectance values obtained are relative values and, for non-fluorescent samples, are independent of the quality and quantity of the light used to illuminate the sample. (Colourware, 2009)
2.6.5 Colour Data File: .qtx file

The .qtx file format (which means the file with ‘.qtx’ as file extension) is the most common file format used for the electronic communication of spectral data for colours. The file contains reflectance data for the colour and additional details such as the tools and procedure used to measure the reflectance data of a colour.

Initially, the .qtx file format is a colour data format used by major vendors of colour calibration hardware, Datacolor products for recording and transferring colour information between different colour management tools (Integrated Colour Solutions, 2005). Now, it has become the widely accepted standard colour data file in the industry.

The file is produced by measuring the reflectance data for a target shade using a spectrophotometer and is formatted by the associated colour management software tool. Figure 2.6 shows a sample .qtx file.

Figure 2.6: Sample .qtx format file tools (Integrated Colour Solutions, 2005)
2.7 Summary

This chapter discusses current practices adopted by most of the companies involving in the colour communication process of textile industry as well as current existing electronic colour communication applications which used by textile industry practitioners in facilitating their daily colour communication process.

There are several major software vendors that produce, market and support software applications that use to automate and digitalize colour communication process in textile industry such as Datacolor, X-Rite and eWarna. However, these proprietary products are often too costly for small players in the industry. Moreover, most of the existing colour communication applications in the market are tightly coupled and platform dependent softwares in which lack of interoperability, reusability, and flexibility. Hence, when a textile industry organization desires to purchase a new piece of product in order to accommodate certain functionalities that are currently missing from its daily operation, the organization might find that the newly purchased application is very difficult to be integrated with currently existing applications that were built on different technology platforms.

In order to explore for ideas that could contribute to overcome the challenges faced by the textile industry players in colour communication process, the researcher has further studied the concepts of the possible technology solution, which is Web services technology. Web service architecture, Web service stack and Web service implementation technologies such as XML, SOAP, WSDL and UDDI had been discussed in this chapter. As Web services technologies are supporting open standards, exhibiting the nature of loosely-coupling, platform independent and programming language neutral, the researcher has chosen to develop a colour communication application for textile industry in Web services.
Besides, several colour physics topics such as visible light in electromagnetic spectrum, illuminants and CIE standard observer were also being explored and discussed in this chapter. The knowledge in colour physics is particularly important for the researcher to design the processing logic for the proposed Web services.

In this chapter, the researcher also outlines the development technologies which will be used to develop the proposed Web services. These development technologies include Java Programming Language, MySQL database system, Apache Tomcat Web Server and Apache AXIS Framework.