December 2001

Workflow Standards and XML

Randall Perrey
Brunel University

Mark Lycett
Brunel University

WORKFLOW STANDARDS AND XML

Randall Perrey
Department Information Systems and Computing
Brunel University
randall.perrey@brunel.ac.uk

Mark Lycett
Department Information Systems and Computing
Brunel University
mark.lycett@brunel.ac.uk

Abstract

The paper outlines the significance of the workflow paradigm in the context of engineering business processes. It reviews the standardisation efforts in the field and considers related XML technologies with respect to their relevance to messaging in workflow. The requirements of a range of stereotypical workflow situations are examined along with broad technological approaches to implementation. A case study is then used to illustrate the 'reality' of standards implementation in the commercial products of Independent Software Vendors. This is followed by suggestions for critical application of some all-purpose XML technologies under these circumstances. Broadly the findings are twofold. Firstly the workflow consortia standards are rooted in applied design but subsequent, more widely applicable standards are likely to 'outgun' them. Secondly the standards effort in workflow is driven by the large software vendors in the field at the behest of large customers while smaller, independent vendors see them as additional overheads and restrictive.

Introduction

Business processes are concerned with the commitments, contracts, bargains and promises that people and organizations make, and the consequent actions that ought to happen to accomplish them. These actions include the coordination of personnel, information and material resources (Medina-Mora, Winograd et al. 1992). In general however, real world processes are messy and therefore difficult to model faithfully. Consequently traditional approaches to business process management are best regarded as 'a network in which a number of roles collaborate and interact to achieve a business goal' (Ould 1995).

Workflow technology is appropriate when configuring an information system in line with the business process approach. One result of the engineering and re-engineering of business processes is the promotion of *coordination* as a critical principle in the study of workflow within organisations (Winograd and Flores 1987). To workflow orient a business process involves (a) integrating people and technology in a formally constructed task and resource dependency structure and (b) managing a dynamically configurable instance of this each time the process is run. These two points relate to what it is to do workflow but not how. Coordination based workflow systems provide a framework within which the emphasis is not on optimising any given flow but on supporting the completion of a human process. Coordination based workflow approaches are used to facilitate Enterprise Application Integration, Enterprise Resource Planning and Business Process Re-engineering.

Workflow technology has grown from disparate roots and consequently there has historically been little consistency or interoperability between products. Coordination based workflow approaches demand that workflow engines inter-operate with a wide range of system applications and flexibly facilitate human user interaction. To be commercially competitive they will need to successfully operate in a distributed environment which includes mainframe and client-server, alongside other workflow engines, controlling some software tools and interfacing with others. They must be able to scale gracefully and cope with many users via a variety of disparate media.

To achieve interoperability necessitates standards and the workflow community has responded with industry consortia such as the Workflow Management Coalition (WfMC) and Workflow And Reengineering International Association (WARIA) who identify and clarify common issues and manage development of standards where feasible.

The remainder of this paper is structured as follows. The paper looks at the standardisation efforts in the workflow arena and their rationale. It introduces the Workflow Reference Model, its purpose and main deliverables. Some eXtensible Mark-up Language (XML) technologies are summarised on their applicability to the workflow arena. The case study introduces an independent software vendor (ISV). Some stereotypical workflow scenarios are examined with respect to their requirements and broad technological approaches to implementation are suggested. The actual approach that the ISV took in its efforts to XML enable its product is detailed along with suggestions for alternative approaches.

Standardising Workflow Technology

A workflow management system can provide support in three functional areas:

- Modelling the workflow process at build time. This is closely aligned with business process modelling. Most products will allow some form of simulation of the proposed process. There has been no standardisation agreement regarding modelling methods or simulation techniques although both are forms of CASE model and there are wider standards relating to model transfer between CASE tools.
- Managing the sequencing and routing during runtime. This involves running many individual instances of a process. This is the internal mechanism of the engine and regarded as a key differentiator of products.. There is no appetite for standards in this area.
- Interacting with users and resources during runtim. This is to do with interoperation and interface definition so most of the standards efforts have been directed here.

WfMC Workflow Reference Model

In January 1995 the WfMC devised the Workflow Reference Model (Hollingsworth 1995), in an attempt to establish a common architecture for workflow technology. It consists of a framework identifying characteristics, terminology and components from which separate working groups have been set up to develop detailed specifications. Hollingsworth says of the reference model that it "describes a common model for the construction of workflow systems and identifies how it may be related to various alternative implementation approaches." (Hollingsworth 1995). The reference model identifies general interfaces that might be standardised.

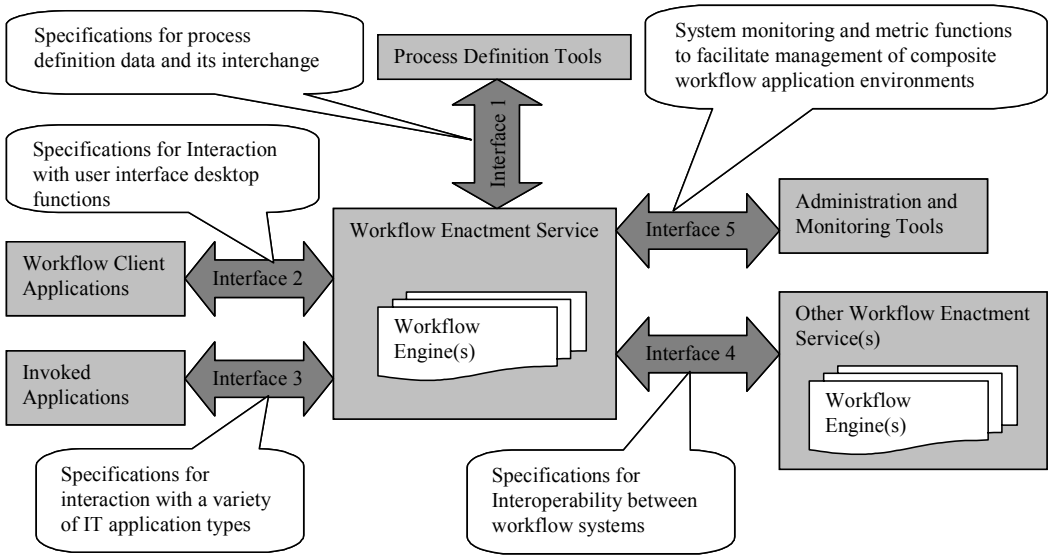

Figure 1. WFMC Workflow Reference Model Interfaces (see references for Specifications and Bindings)

The working groups set up to address each of these interfaces have all produced detailed specifications. In some cases the working groups have worked with partners to produce bindings in some technology. The OMG binding for the Client Application Programming Interface and Workflow Interoperability have been combined and extended to include all runtime interactions and create one Workflow Management Facility IDL specification endorsed by both the WfMC and the OMG (Workflow Management Facility Specification, V1.2 - also know as the **JointFlow** Specification).

XML and Related Technologies

Each of the interfaces of the reference model can be addressed using XML and related technologies. There are a number of advantages in the application of XML. Firstly, it is possible to separate the messaging protocol from the communication medium thus lifting any proprietary restrictions or limitations on the environment within which the product can work. Secondly, XML is easily transformed from one syntax to another making it a flexible protocol for interactions in environments with heterogeneous data prescriptions. Thirdly, XML is open and there are already many parties that are actively engaged in providing transportation, interpretation and transformation mechanisms for almost every conceivable situation. Fourthly, the extensible nature of XML offers some protection against obsolescence as messaging protocols can be defined in schema that are referenced in each message making XML a self describing protocol. Developing a layered technology architecture on top of an extensible and simple messaging format provides robustness, simplicity, reusability and interoperability. With all these advantages and more there are many competing and complimentary technologies being developed.

Within the workflow community the Simple Workflow Access Protocol (SWAP) was proposed as a binding of the JointFlow specification to an HTTP - XML based protocol building on the WebDAV extensions to HTTP (Hayes, Peyrovian et al. 2000). This protocol subsequently evolved into Wf-XML which is an application specification designed to define how the XML language is used to communicate workflow-related processes and data between different workflow applications. The WfMC abstract specification defines *'what'* needs to pass between workflows, Wf-XML specifies *'how'* to use the XML language to make it happen (Rogowski and Rossi 2000). Wf-XML requires a highly structured XML message to support the protocol which does not address the data syntax, leaving this to a separate, appropriate domain data schema. Divorcing the transport mechanism from the messaging protocol removes the functional issues of each interoperation scenario (connectivity, security etc.) into a separately negotiated interoperability contract. One issue generated by this is that it inhibits the extensibility benefits of using XML and negates the possibility of self-describing messaging interpreted on the fly. For most scenarios this is unlikely to be a problem but allowing for the provision of referenced schemata in the messages relating to both the Wf-XML and the data syntax schema would reintroduce the possibility.

The Simple Object Access Protocol[1] (SOAP) is the latest and most well supported evolution of XML-RPC messaging since it is incorporated into Microsoft's Biztalk technology. SOAP can be used over different transport protocols such as HTTP, SMTP or IIOP. Its messaging methodology allows for its use as a response/request remote procedure call (as defined by the Open Group's[2] Distributed Computing Environment (DCE) specification), or one way message passing, for example into a queuing system for batch style processing. SOAP has the payload XML message in an envelope and all the variations and capabilities that apply to standard XML apply to it.

The World Wide Web Consortium (W3C) has established a working group to study the area of remote procedure using XML, called XML Protocol. Amongst other factors this follows the submissions of SOAP for consideration. This marks a major step for the W3C towards developing XML technologies which allow software peers to communicate in a distributed environment.

The initial focus of the XML Protocol Working Group is to develop a framework for XML-based messaging systems, which includes specifying a message envelope format and a method for data serialization, directed mainly, but not exclusively, to RPC applications.[3] The developments within the XML protocol working group are almost certainly going to be of significance to the WfMC and most likely will impact on the development of future releases of Wf-XML.

Case Study

The case study was of an Independent Software Vendor (ISV) making its first development attempts using XML. The company's investment capital was mainly in one workflow product. The product is component based which enables it to be embedded in other vendor's software products to provide them with workflow capability.

---

[1]URL: http://www.w3.org/TR/2000/NOTE-SOAP-20000508 (01/09/2000).

[2]URL: http://www.opengroup.org/onlinepubs/009629399 (01/09/2000).

[3]URL: http://www.w3.org/2000/xp (15/09/2000).

The Vendor wanted to sell its product to companies whose business is the supply of computing services to third party clients. This industry operates either of two basic models. One model utilises the web with companies supplying entire computing infrastructures to third parties who have nothing but web browsers and internet connections. In this situation the workflow product invokes applications locally but co-ordinates human resources via a browser over the web.

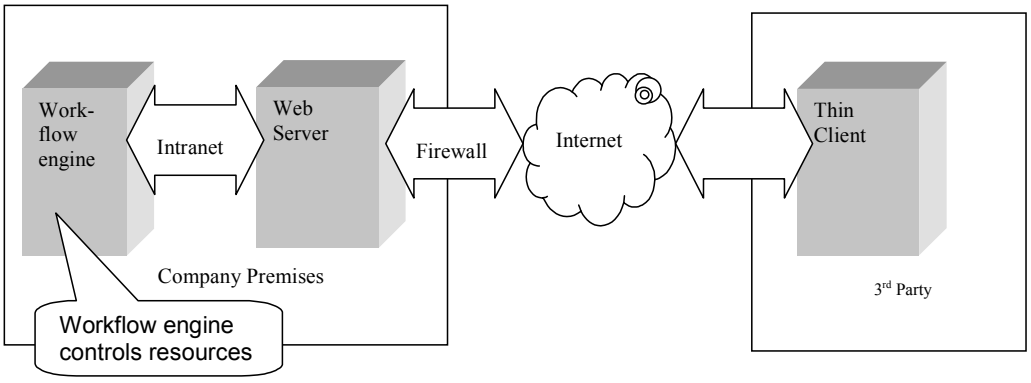

Figure 2. Generic Structure for a Thin Client Application

In the other model the companies supply consultancy services and components to integrate and automate a third party's business processes. Often, these companies provide maintenance for the resulting system and update or augment as necessary. In such a model the workflow product would need to operate, monitor processes and manage applications distributed over an Extranet or possibly the Internet.

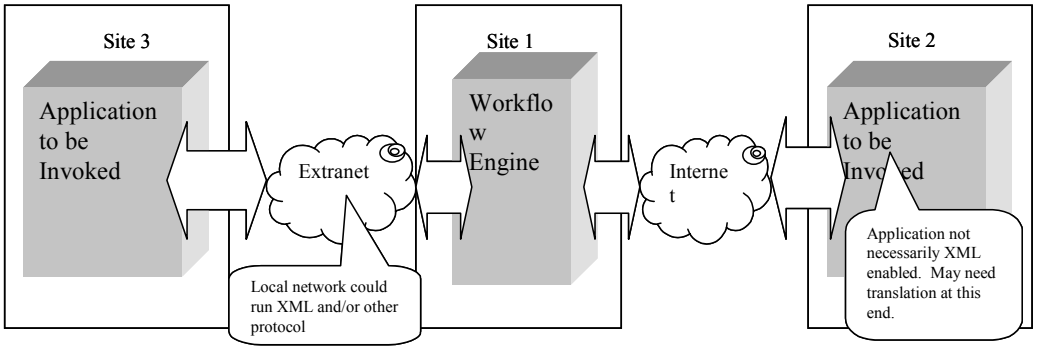

Figure 3. Generic Structure for Invoking an Application Using XML

Background

The Product is built according to the Component Object Model (COM) and its network extension, Distributed Component Object Model (DCOM). Vendors that wish to embed the product in their own will need to accommodate its COM heritage. The decision was made to upgrade the existing product and maintain the existing code, which represented such an investment of time and effort and was well tried and tested. The vendor wanted to leverage the COM interfaces it already had so the development work centred around wrapping the existing engine in XML to COM translation code. The vendor assumed the use of eXtensible Styling Language Translation engine (XSLT) to turn any other data format into the appropriate XML syntax for their wrapper code.

Each parameter of the COM interface needed a corresponding XML element defining parameter constraints such as data type and similarly since each method had a number of parameters it also needed its own schema defining the tag set and their allowable structure. The generic COM to XML translation class was written to accept the set of parameters it was given from either incoming or outgoing messages and translates each element while internally maintaining the structure of the overall message. The engine component interface was known and immutable thus allowing a fixed conversion mechanism for outgoing messages and avoiding the need to reference a schema. This was a deliberate constraint placed on the design after tests indicated that outgoing messages were comparatively large and validating them would considerably reduce performance. Incoming XML however could not be assumed to be compatible so the translation engine was written to validate the incoming message against a schema to check for correct structure and establish the element types before applying the appropriate conversion to the associated COM parameters for the method call.

Use of XML in Thin Client Environment

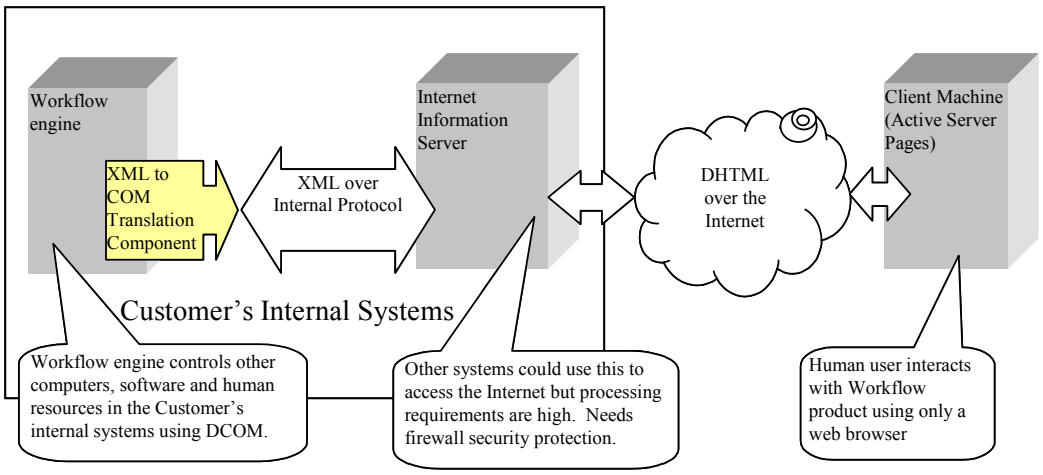

Figure 4. Actual Use Of XML in Thin Client

The Vendor chose not to use XML as a way of remote messaging across the Internet. XML is used as a way of interacting with the web server in such a way that a Dynamic HTML page is created for consumption on the client browser. The existing workflow engine with its generic wrapper translates the native COM calls into XML. Local applications are still addressed using COM calls. The XML passes over the internal network between the workflow engine and the web server, which must be Microsoft's Internet Information Server (IIS). The perceived ultimate process is that the XML is pushed out across the Internet to XML enabled browsers accompanied by style sheets. For the time being this arrangement is not in use. The current system is that the web server transforms the XML into DHTML and sends this over the Internet and the clients interact from their web browsers using Active Server Pages (.asp). The Active Server Pages contain scripts that alert the web server when events happen on the client browser, such as the client clicking a button. The use of ASP takes advantage of the in-built firewall negotiating capability and scripting alert mechanisms. The browser event alerts are translated on the web server into XML with the appropriate syntax deduced from the schema and passed back over the internal network to the wrapper code and from there into native COM calls for the engine. This is illustrated by figure 4.

Incoming XML messages are currently formed into a tree hierarchy using the Document Object Model (DOM) and validated using a schema. The schema not only allows the DOM to check that the message is correctly structured but also specifies the data types and constraints thus giving the translation engine the information it needs to convert to a COM method call. In order to validate against a schema the appropriate schema must be referenced in the message.

Internet clients interact using DHTML generated by the IIS server and therefore cannot be expected to reference the schema of the call they are making. The client DHTML is translated to XML by the IIS server which must select and add the appropriate schema reference to the XML document. The schemata are located on the IIS server and the path to the schema is stored in the registry of the workflow engine machine so that the translation engine can find them. The conversion of outgoing XML to DHTML is hard coded into the Active Server Page along with the necessary code for the construction of the workflow interface and the interaction mechanism. Additional translations incur a substantial performance hit and some considerable effort was invested in accelerating this process. Using XML throughout would remove the need for such translations but the vendor undertook market research at a time when Internet client browsers could not deal directly with XML Modern Internet client browsers can utilise eXtensible Style Sheets to interact directly in XML and eXtensible Style Sheet Translation to convert from one form of XML to another eg. Wf-XML. An enabled browser could provide an interface that allow users to link to other resources in the enterprise (also XML enabled) so that having been alerted to the task at hand by the workflow engine, they have immediate access to the resources to carry it out. An added benefit of developing in XML would be to remove object model dependency and so utilise the development potential of more than one platform vendor.

Mechanism for Invoking Applications

The vendor does not use XML technology to directly invoke applications remotely. The current mechanism is to distribute the workflow engine over machines in each site using DCOM and invoke applications using the locally available engine.

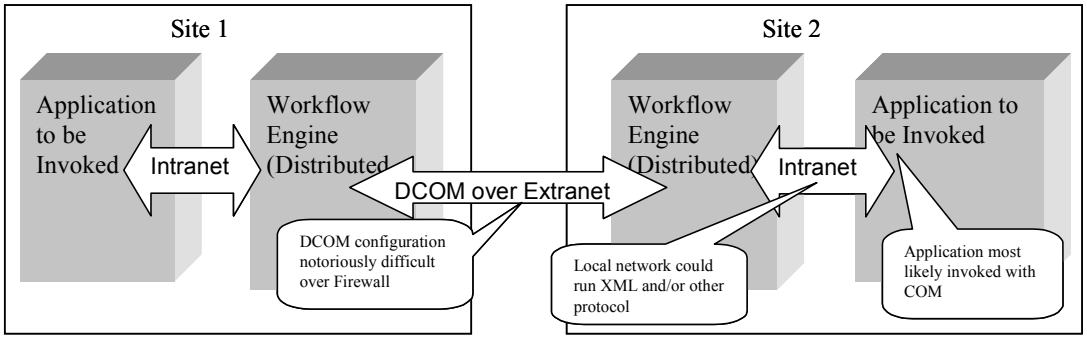

Figure 5. Actual Mechanism for Invoking an Application

The formation of the incoming XML into a tree structure in memory using the Document Object Model (DOM) is a processing and memory heavy mechanism for translating into a COM call, especially given the potentially high volumes of incoming messages. The DOM is a mechanism designed to allow manipulation of the tree structure but this is a facility that is rarely needed when applications communicate. Using an event driven parser model would be significantly faster. It could be primed with the trigger of reading the tags for each parameter and 'fire off' the events of passing the data held within on to the translation section of the engine. The Simple API for XML (SAX) is just such a parser. One thing that the DOM does do but SAX does not currently do is handle datatype translations. For instance, Boolean datatypes, which differ in XML and C++, would need to be addressed by an additional mechanism, perhaps within the translation wrapper code.

The current mechanism is conceived with extensibility in mind, to accommodate future changes in XML technology. The assumption is that the development work is the first step on the road to being more fully XML enabled. The decision to stay with the proprietary technology but XML enable it is not only unnecessary but counterproductive. The most significant design limitation is performance and it is the efforts to maintain extensibility in the legacy mechanism that limits it. Using a fixed vocabulary XML would remove extensibility and collaboration opportunities but would be faster to process. If the workflow engine only has to interpret XML generated automatically in the correct syntax then the COM - XML translation speed can be increased. There is no necessity to validate against a schema if each tag is used uniquely, in a one to one, tag to component way, and each element has an obvious and predetermined type. An element set that is predefined and not referenced is using XML as just another network protocol. It is a limited way to use XML but it has advantages. A fixed syntax would make it possible to invoke applications using the XML over the Internet through an equivalent translation wrapper running on a remote machine. It would facilitate remote monitoring and auditing as necessary.

Conclusions

This independent software vendor has in fact invested little in development costs towards XML enabling their product and predicts that the customer base currently using the design will account for the return on this investment. In a sense then this development work has been disposable. For the future alternative solutions should be sought that address the problems with greater elegance. If additional development work is undertaken along the current lines the result may well become a legacy system.

The standards that were obviously needed to unite and stabilise the workflow industry are well advanced and there is some cross-pollination with standards efforts in complementary fields. For independent software vendors however, the business case for developing standards into working prototypes is far from clear. Most are not in a position to incorporate into their schedules development of non - commercial implementations just to reflect new approaches. Such blue-sky research and the funds necessary to resources it, is still the preserve of the major software players.

If the independent software vendor had considered the long term view it might have built next generation workflow components with Wf-XML as the native messaging protocol. This would maximise the flexibility for deployment purposes while minimising the performance hit when issuing and receiving messages by saving on transformation. The specific problems and obstacles outlined in the case study are illustrative of the competing business demands of short term goals verses long term aims. The study highlights the strategy of concentrating on utilising technology to improve existing products for immediate sale. The latest workflow standards and approaches are not perceived as the drivers for current business situation. Ignoring the direction that the market is taking however, is likely to lead to obsolescence in the long term. An intermediate approach should be sought that offers the opportunity to derive short-term commercial benefit while facilitating longer term redevelopment in line with the evolving workflow standards.

References

WfMC Interface Working Group 1 - WfMC-TC-1016-P V 1.1, IWG 2&3 - WFMC-TC-1009 - Specification V 2.0, Naming Conventions - WFMC-TC-1013 V 1.4, IWG 4 - WfMC-TC-1012, 20-Oct-96, 1.0, IWG 5 - WFMC-TC-1015, 22-Sep-98, 1.1

WfMC / OMG IDL binding Workflow Facility Specification, Draft WfMC TC-2101

WfMC Internet e-mail MIME Binding WFMC-TC-1018, 7-Jan-00, Version 1.2

WfMC Asynchronous HTTP Binding of Wf-XML WfMC-TC-1023, 8-May-00, 1.0

Hayes, J. G., E. Peyrovian, et al. (2000). "Workflow Interoperability Standards for the Internet." *IEEE Internet Computing* **4**(3): 37-45.

Hollingsworth, D. (1995). Workflow Reference Model. Winchester, Workflow Management Coalition.

Medina-Mora, R., T. Winograd, Flores, Fernando Flores, Rodrigo. (1992). The Action Workflow Approach to Workflow Management Technology. *Conference Proceedings on Computer-Supported Cooperative Work,* Toronto, Ontario Canada, ACM.

Ould, M. (1995). *Business Processes.* Chichester, John Wiley.

Rogowski, J. J. and M. Rossi (2000). What is Wf-XML and What does it Mean to Your Business?, Workflow Management Coalition (WfMC)

Object Management Group (OMG)

Black Forest Group (BFG). 2001.

Scherr, A. L. (1993). "A New Approach to Business Processes." *IBM Systems Journal* **32**(1): 80-98.

Winograd, T. and F. Flores (1987). *Understanding computers and cognition : a new foundation for design.* Reading, MA: Addison-Wesley.