MedBiquitous Virtual Patient Architecture

MedBiquitous Virtual Patients (MVP)

An Architectural Model for MedBiquitous Virtual Patients

Draft White Paper
Version 6: September 11, 2006

Rachel Ellaway, Chris Candler, Peter Greene and Valerie Smothers - MedBiquitous Virtual Patient Working Group (with additional input from the MedBiquitous Technical Steering Group)

1: General Principles

MedBiquitous is developing a data standard for the exchange and reuse of virtual patients. This will be the MedBiquitous Virtual Patient (or MVP) standard. A working definition of a virtual patient is:
An interactive computer simulation of real-life clinical scenarios for the purpose of medical training, education, or assessment. Users may be learners, teachers, or examiners.

This document outlines a proposed structure and possible implementations of the MVP standard currently under development. The proposed structure allows for a wide variety of educational approaches and technologies. The appendix provides some exemplar use case scenarios, including the following:

- Learners access virtual patient data and associated media resources in a self-directed fashion.
- Learners participate in a problem-based learning activity which includes concomitant virtual patient data.
- Learners access a longitudinal, activity-based virtual patient with integrated media resources and collaborative learning tools.
- Learners participate in a multi-player game with integrated learning activities, virtual patient data, and media resources for each case.

2: Proposed MVP Architecture

There are many different ways that virtual patients can be created and employed. The MVP data standard must therefore be sufficiently abstract and adaptable so that it can accommodate a number of forms and uses.
The proposed architecture consists of five MVP components (see section 3), each of which can be accessed and assembled in a number of different ways (see section 4). These components are rendered for use through different kinds of players, depending on the activity at hand and other local choices and requirements. These five and other related (but out of scope) components such as user input and feedback, student profiles, and tracking are shown in figure 1.

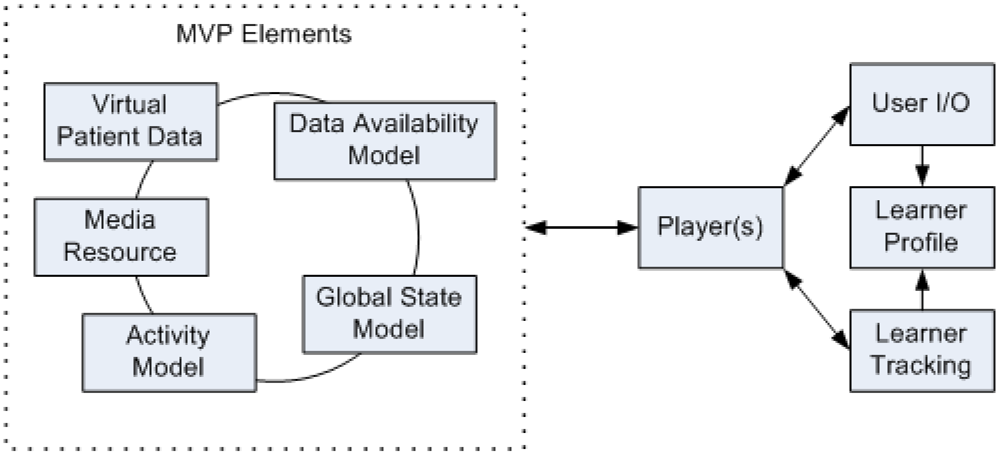

Figure 1: components of the MVP architecture

3: MVP Components

There are five core MVP components:

1. Virtual Patient Data (VPD)

The VPD provides the personal and clinical data that is relevant to the clinical scenario being simulated. The VPD is a bit like a clinical chart, being divided into sections that correspond to the medical history, physical examination, laboratory and radiology data, and procedure and outcome data.

An XML model is being developed for this component that enables a flexible approach to the data. Data within each section is tagged with identifiers so that patient data can be disclosed progressively. The data can be disclosed to the learner en bloc, or the data can be structured so that it can be disclosed iteratively in response to specific learner requests, with the XML allowing the encoding of both the request and the response information as is logical for each section:

- Medical History: questions and answers
- Physical Examination: examinations and finding

- Laboratory tests: tests and results
- Radiographic tests: tests and results
- Procedures: procedures and outcomes

Each of the sections may include subsections that further refine the information available. The Medical History section, for example, may include subsections for present illness, past history, family and social history, etc.

Most of the VPD data is provided in unstructured text, but certain sections may be highly structured and encoded with healthcare terminologies (RxNorm for medications and SNOMED for allergies, problems, and procedures). If this encoding is present, it could enable the learner to interact with clinical decision support systems during the virtual patient learning activity.

2. Media Resources (MR)

Media resources are all of the images, animations, videos, and audio files that are associated with the virtual patient at any point during the simulated patient scenario. As with the specific portions of the VPD data, the media resources are tagged with identifiers so that they the can be made available at the right time to the learner. IMS Content Packaging will be used to catalogue media resources and provide unique identifiers.

3. Data Availability Model (DAM)

This component enables the sequencing and progressive disclosure of data encoded in the VPD and associated media resources. The Data Availability Model references the identifiers within the VPD and sites specific media resources to allow for the targeted or progressive release of parts of the VPD or individual media resources.

Note that the state model defines whether the consuming player application is cumulative (items from lower state values remain available) or selective (only items within the current state are available). The state model also allows for the encoding of non-linear or branching activities, with user decisions (or indecisions) determining when and which state is reached next.

The Data Availability Model exists separately from the VPD and media resources components because it is quite possible that the same underlying clinical scenario could be used by very different player applications, each of which has a Data Availability Model that encodes the appropriate level of complexity.

4. Activity Model (AM)

This component encodes how the learner will be able to engage with the virtual patient. A number of different learner activities are possible with the same underlying virtual patient dataset, and the activity model component encodes the activities available, often including the narrative context. Examples of activity models are:

1. **Observation:** The learner is presented with a 'page turner' and is allowed to observe how a clinical scenario unfolds. (This might be appropriate for beginner students that are not ready to make interventional decisions).
2. **Free Navigation:** The learner is presented with clinical materials which are linking to a variety of references (journal articles, books, data sets, decision support systems, etc) that they are free to navigate in an unrestricted fashion.
3. **Diagnosis Formulation:** The leaner is asked to formulate a differential diagnosis at each stage of the progressive disclosure of the clinical scenario.
4. **Decision Making:** The leaner is asked to make clinical decisions based on current information and acquires new information as a result of different decisions.

For each of these activity models, the corresponding Data Availability Model must be designed in concert to support the activity, and these two components will typically work closely together. For example, the observation activity would work with a Data Availability Model that was cumulative and linear, whereas the decision making activity would require a Data Availability Model that is selective and branching.

Different players (see below) will need to interface with the VPD, Data Availability Model, and media resources components, but not necessarily all of them. It is a fundamental requirement of the MVP standard that VPD and activity model components are able to function independently of each other. Thus a paper case could be derived directly from a VPD and an activity based on clinical decision pathways could be designed without direct dependence on a particular VPD data set.

5. Global State Model (GSM)

This component provides the top-level modelling of student activity and the logic for the allocation and regulation of activities. It is only relevant for the more advanced kinds of applications; for example, it might provide information about how a particular virtual patient interacts with other virtual patients in a multi-player educational game.

4. External Components

In addition to the five MVP components, there are several applications and data sets that are related but external to the MVP standard:

- **Player:** This is the application that presents the virtual patient to the user and gathers and processes user input. It is anticipated that there will be many different kinds of virtual patient players, depending both on the kind of user activity or rendering required and on the local online learning environments used. A virtual patient player could run inside a Web browser and be coded in Javascript or Flash (such as the University College of Dublin system), or the player could be a server-based application (such as the systems developed at the Karolinska Institute, Harvard Shapiro Institute, Tufts, Pittsburgh, NYU), or the player could be an online game (as is being developed at the University of Edinburgh).

- **Learner Tracking:** This component captures and stores interactions and performance metrics about the learner from the player component. Tracking data could be added to a learner profile or provided as de-briefing feedback for a learner.

- **Learner Profile:** This component is a dataset of information about the user. It includes two kinds of information:

-

- General user information: user identity data (such as their name), as well as their current courses (or programme of study), personal user preferences such as choice of language, required disability support features or choice of stylesheets, and academic data such as assessment data.
- Virtual patient usage information: tracking data, records of which virtual patients have previously been accessed (including progress markers) and their relation to study markers such as learning objectives and outcomes.

5: Levels of MVP Complexity

Section 2 of this document described the overall MVP architecture and section 3 enumerated its components. This section describes how the components are related and how they can be connected to create a learning experience.

The principle behind the MVP architecture is that it is highly adaptable and configurable to different kinds of settings and applications. These have been classified according to their complexity in levels 1-4, each of which is described in turn.

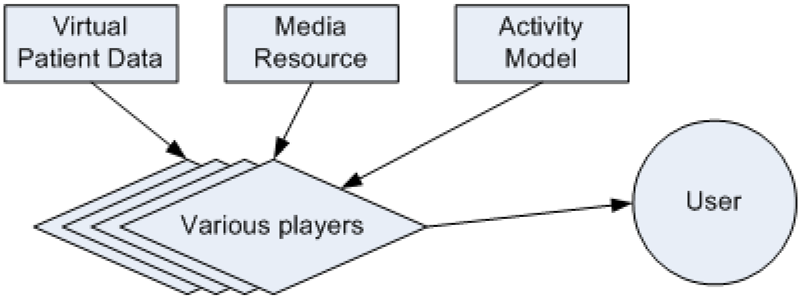

Figure 2: Level 1: MVP components are accessed directly with no dependency on one another. In this example, patient data, supporting media resources, and activity model data can be accessed through the player without any interrelationships between the components.
(empty line)

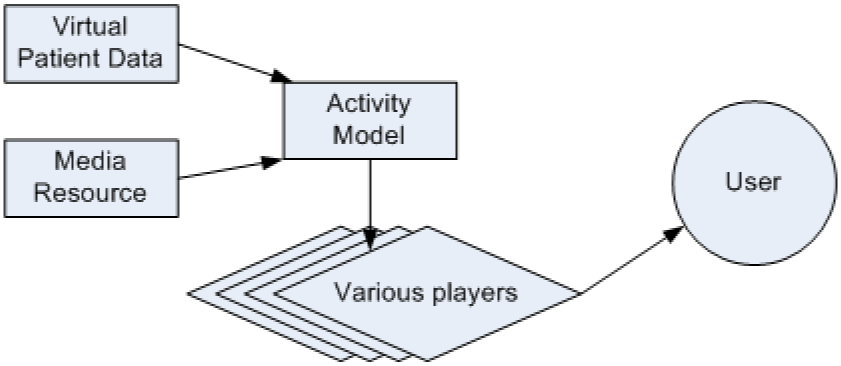

Figure 3: Level 2: In this level, MVP components are (statically) referenced through others so that they appear as a coherent whole. In the example above, virtual patient data and media resources are referenced within an activity model.
(empty line)

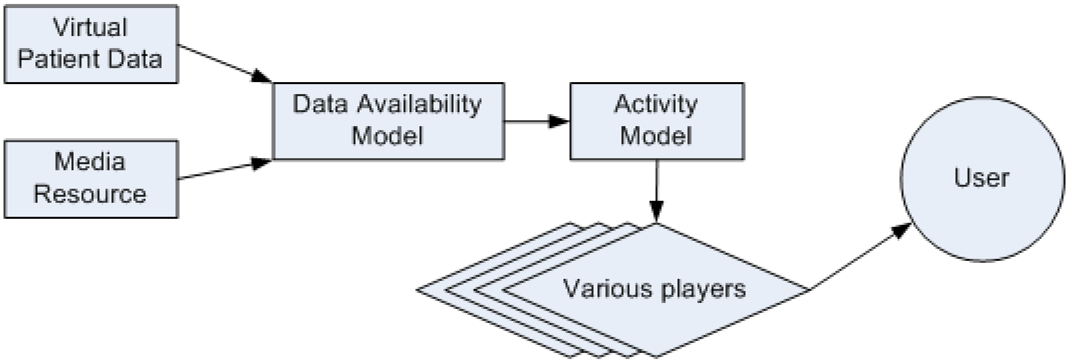

Figure 4: Level 3: Some components are (dynamically) referenced through others so that they appear as a coherent whole. In this example patient data and media resources are referenced in an activity model accessed through a Data Availability Model.
(empty line)

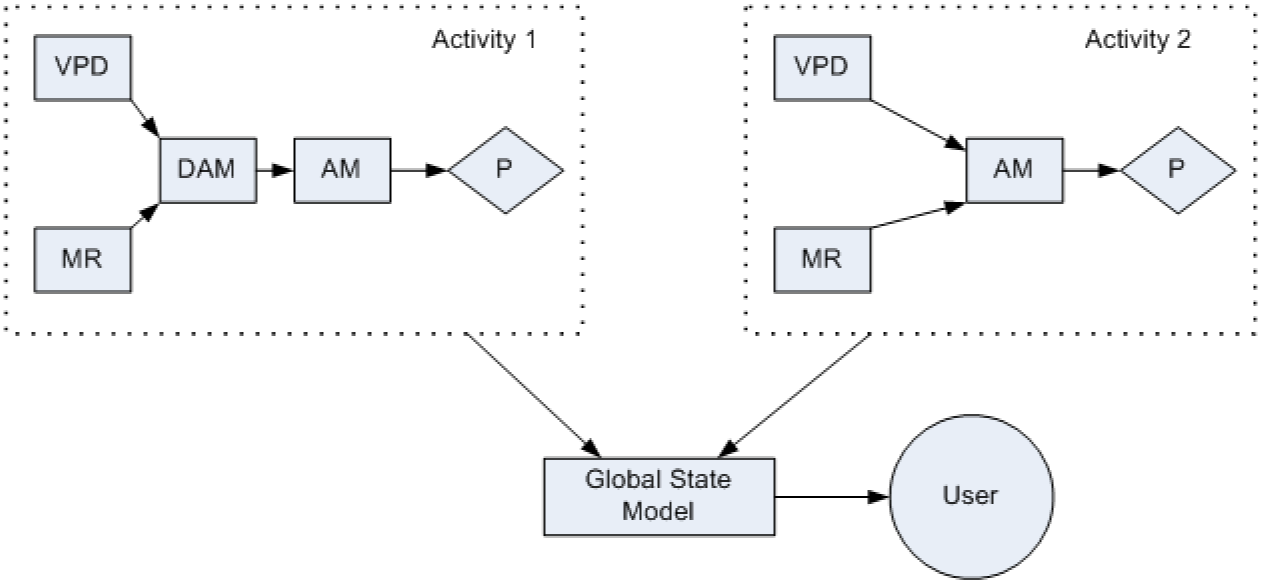

Figure 5: Level 4: Any number of Level 1, 2 or 3 structures are coordinated by a global state model and rendering mechanism. In this example two separate Level 3 activities are referenced through a single global state model.
(empty line)

Virtual patient applications at specific complexity levels can be created in a number of different ways. A number of examples are listed in table 1 with links to their respective use case scenarios in the appendix.

Level	Components	Description	Use Case
1	VPD	Just the virtual patient personal and clinical data	1
1	MR	Just resources representing the virtual patient	1
1	AM	Just the virtual patient activity (such as 'clerk this patient')	-
2	DAM + VPD	Personal and clinical virtual patient data organised into state levels and thereby set up for filtered release.	-
2	DAM + MR	Virtual patient media resources organised into state levels and thereby set up for filtered release (such as video clips outlining the progression of a consultation).	-
2	DAM + AM	Activity components organised into state levels and thereby set up for parsing. For instance parts of an activity only become available once other parts have been completed.	-
2	AM + VPD	Activity model drawing upon VPD data	2
2	AM + MR	Activity model drawing upon resources	-
3	AM + DAM + MR + VPD	Activity model drawing upon VIRTUAL PATIENT data and media resources via Data Availability Model (DAM) levels	3
n	GSM + AM + DAMs + VPDs + MR	A series of activities governed by a global state model (GSM) - each activity may be level 1, 2 or 3.	4

Table 1: different kinds of complexity level examples with links to appropriate use case scenarios.

6: Interoperability and Runtime

Originally, standards like the one in discussion here were intended merely to provide interoperability between different systems. However, while this pertains for non-dynamic areas such as metadata and learner profiles, it is less so for dynamic areas, in particular those that encode or are related directly to human activity, of which IMS Learning Design is a good example. In this latter case there are implications for both interoperability and runtime.

This section is provided as an indication of how MVP components could be instantiated in runtime environments and uses a number of UML sequence diagrams to show how different components in different kinds of applications communicate and interoperate with each other.

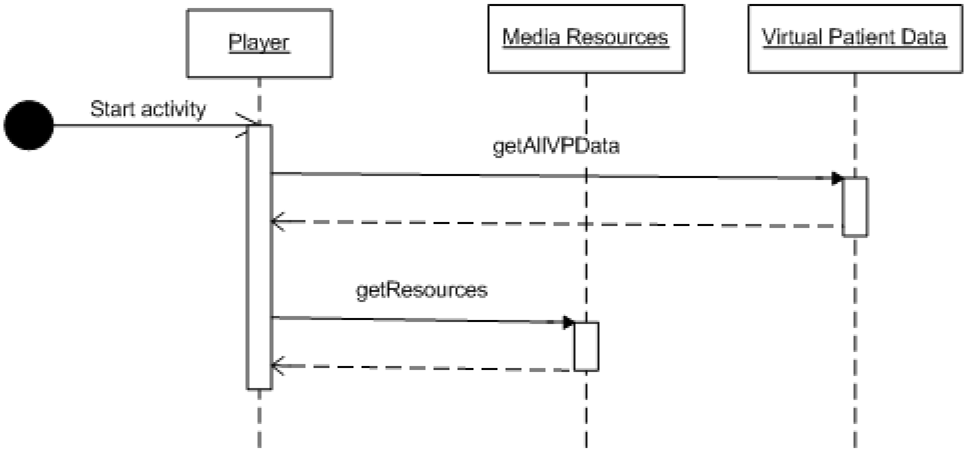

Figure 6: a Level 1 application where VPD data and media resources are accessed directly by the player (see use case scenario 1 in appendix).
(empty line)

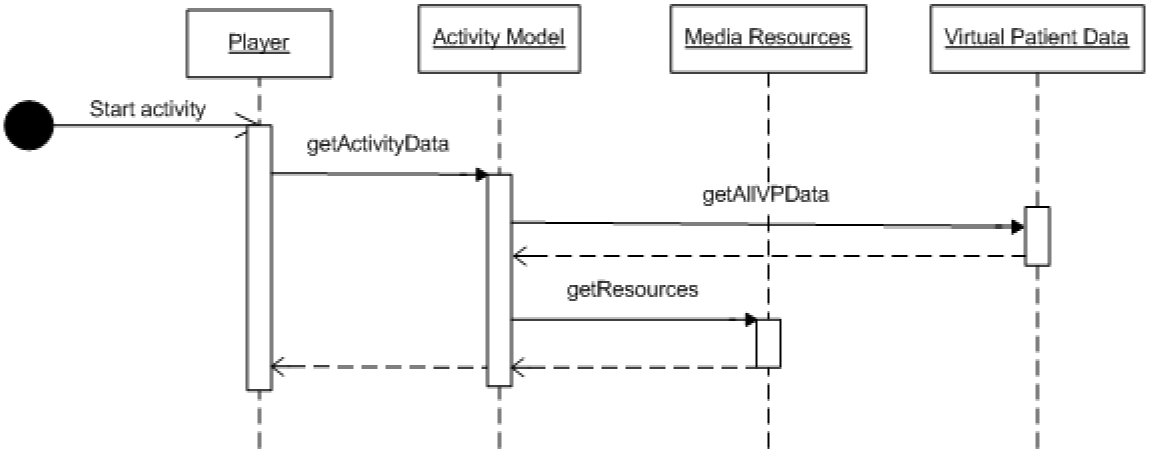

Figure 7: a Level 2 application where VPD data and media resources are accessed via an activity model (see use case scenario 2 in appendix).
(empty line)

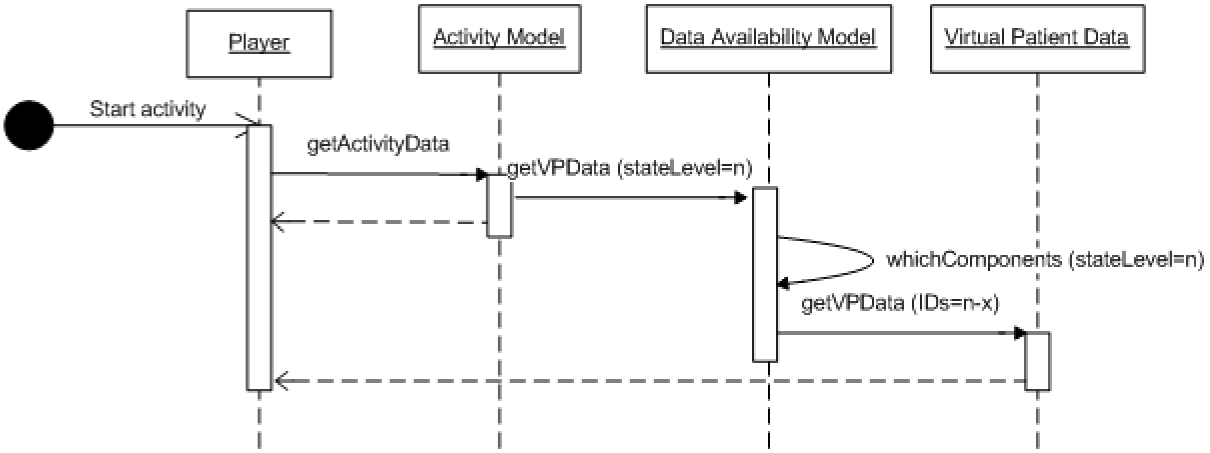

Figure 8: a Level 3 application where VPD data is accessed by an activity model via an intermediary Data Availability Model (see use case scenario 3 in appendix).
(empty line)

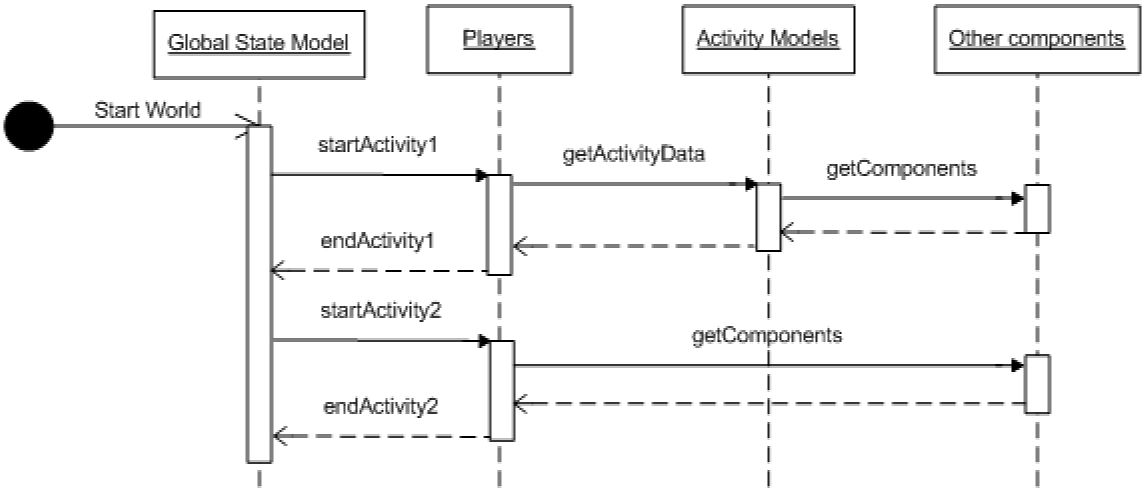

Figure 9: a Level 4 application where a global state model first selects an activity model that accesses other MVP components, then a second presentation of components alone (see use case scenario 4 in appendix). Note that getComponents is a simplified version of the different kinds of Level 2 and 3 applications described earlier.
(empty line)

6: Development Roadmap

It is clear that the proposed MVP architecture will not be developed over night and it should also be noted that the MVP architecture is not being introduced de novo; there are both extant MVP architectures (for instance in most of the MVP working group's participants schools) and extant learning technology and healthcare interoperability standards and specifications. The development roadmap must accommodate these factors.
The proposed development stages are as follows:

- A common Virtual Patient Data model
- Media Resources
- Data Availability Models
- Activity Models
- Inter-component messaging
- Players
- Connectivity and messaging with other systems

The first stage to be tackled will be the development of the VP data model (See Figure 10). One advantage of the multi-component and multi-complexity level approach is that it allows the near-term deployment of simpler MVPs as the standard is emerging.

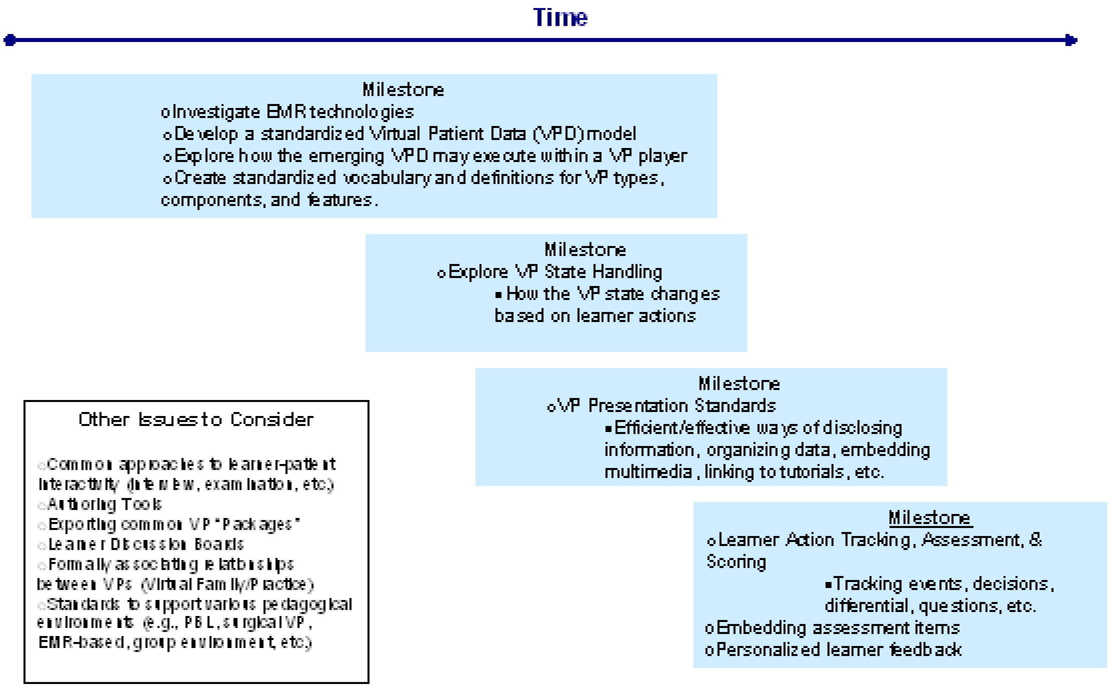
Time
Milestone
o Investigate EMR technologies
o Develop a standardized Virtual Patient Data (VPD) model
o Explore how the emerging VPD may execute within a VP player
o Create standardized vocabulary and definitions for VP types, components, and features.
Milestone
o Explore VP State Handling
▪ How the VP state changes based on learner actions
Milestone
o VP Presentation Standards
▪ Efficient/effective ways of disclosing information, organizing data, embedding multimedia, linking to tutorials, etc.
Milestone
o Learner Action Tracking, Assessment, & Scoring
▪ Tracking events, decisions, differential, questions, etc.
o Embedding assessment items
o Personalized learner feedback
Other Issues to Consider
o Common approaches to learner-patient interactivity (interview, examination, etc)
o Authoring Tools
o Exporting common VP "Packages"
o Learner Discussion Boards
o Formally associating relationships between VPs (Virtual Family/Practice)
o Standards to support various pedagogical environments (e.g., PBL, surgical VP, EMR-based, group environment, etc)

Figure 10: The Virtual Patient Development Roadmap

8: Summary

This white paper has presented a five-component architecture for MedBiquitous Virtual Patient standard that it is anticipated will accommodate many different extant and future applications of virtual patients. A number of different complexity levels have been described that connect and instantiate these five components in different ways.

The purpose of this white paper has been to outline and argue the case for this approach and to demonstrate both its extensibility and intrinsic simplicity. The development of specifications and/or standards for the MVP components and the way they are connected may yet lead to further changes and refinements of the architecture and applications, but the basic principles and approach are expected to persist.

Appendix: Use Case Scenarios

Although there is a strong theme of problem based learning (PBL) running through these use case scenarios it is for illustrative purposes and as a means to draw comparison between the cases. There are many other valid non-PBL uses of virtual patients. For a range of examples see 'Modeling Virtual Patients and Virtual Cases' by Rachel Ellaway in MELD at http://meld.medbiq.org/primers/virtual_patients_cases_ellaway.htm).

ID	MVP-WP-01
Title	VPD + Media Resources
Players	VPD player, resources player, student, tutor, virtual learning environment
Assumptions	Suitable VPDs and resources exist, all users have access to suitable networked computers (or equivalent). A clinical tutor is preparing materials to support a PBL activity.
Description	First of all the tutor checks the available bank of VPDs and finds one that suits the activity she wants to run. She makes a number of small changes to tweak it to her needs. She then searches, locates and assembles a number of media resources (clinical photographs, CT scan images, ECG traces and video clips) to support and augment the VPD data. The VPD (in its XML format) and resources are uploaded to the local virtual learning environment to an area set aside for this particular activity and represented as a 'PBL resource pack.' At the start of the PBL activity the students in the PBL group are directed to download the packaged resource pack for the case from their virtual learning environment. The VPD is loaded into a Flash-based player for viewing as a tabbed set of clinical notes and the resources are accessed via individual hyperlinks from an HTML index page in the virtual learning environment. Some resources such as streaming video are accessed via appropriate media players.
Transactions	Tutor searches and finds VPD and then amends it. Tutor searches and finds resources. Tutor creates a holding space in local virtual learning environment and uploads VPD and resources to create a PBL resource pack. Student accesses virtual learning environment and browses to PBL resource pack. Student loads VPD into Flash-based player. Student views resources using appropriate players.
Exceptions	VPD transformed to HTML and run through a regular web-browser as its player. Different runtime VPD player(s) used.

Notes: in this use case scenario two components of the MVP architecture are employed but not linked. This use case scenario represents a non-coupled, non-dependent model with MVP components acting as distinct artifacts.

ID	MVP-WP-02
Title	Activity Model + VPD
Players	Activity player, VPD player, student, tutor, virtual learning environment
Assumptions	Suitable tools and resources exist, all users have access to suitable networked computers (or equivalent). A clinical tutor is preparing a PBL activity. She wants to create a series of web pages that will show patient data as her students work through the case and deal with emerging problems. She also wants her four different groups to deal with slightly different patient presentations.
Description	The tutor first adapts an existing virtual patient to show the basic information required for the activity. She then makes four copies and changes some details on each such as the patient's name, date of birth, test result details and so on. These are loaded into the local VPD player. The tutor then writes a workbook as a series of web pages (in a courseware authoring system) along with questions that take the student through the essential stages in the PBL activity. The workbook is also copied and loaded into the activity player in the local virtual learning environment and each one is assigned to one of the four PBL groups. The last stage in preparing the activities is to connect each of the different VPD variants (via the VPD player) to the four different group's activities thereby making them subtly different. At the start of the activity groups of students are directed to work through the online activity. The students logon and access the activity as a group through the activity player. They work through the exercises and questions in the activity. From time to time they launch and refer to the virtual patient record in a new activity player window (running the inline VPD player) in order to support their studies.
Transactions	Tutor locates and adapts virtual patient. Tutor makes copies of virtual patient and customises each in turn. Tutor uploads new virtual patients to local VPD player. Tutor creates online activity workbook. Tutor makes copies of the workbook and assigns them to the four different PBL groups. Students access the activity player in groups and work through online PBL workbooks. Students call up the virtual patient record in the virtual patient player from within the activity (workbook) player.
Exceptions	Tutor is supported in the preparation by others (such as learning technologists). Same VPD and workbook used for all PBL groups. Students work alone rather than in groups.

Notes: VPD player needs to have the unique VPD ID passed to it in order for it to load the appropriate VPD data. The VPD player may be a component of the activity player rather than a separate subsidiary system.

ID	MVP-WP-03
Title	Activity model + Data Availability Model + Media Resources + VPD
Players	Tutor, students, computer assisted learning package player, virtual learning environment
Assumptions	Suitable tools and resources exist, all users have access to suitable networked computers (or equivalent). A clinical tutor is preparing a long case PBL activity. She wants to create a series of web pages that will progressively show more and more patient data as her students work through the case and deal with emerging problems.
Description	The tutor first adapts several existing virtual patients to represent the same patient over time with a number of different presentations and interventions. These VPD data are merged to create a single virtual patient suitable for a long case exercise. The PBL exercise consists of a number of presentations and associated questions and problems for the students to work through. These are assembled as a series of steps in a computer assisted learning package with key questions or decisions to structure progress. When each question set has been answered more patient data is released to the student. The tutor therefore assigns different parts of the virtual patient to different sections of the online package. The activity is essentially linear so the tutor sets the release so that data, once released, remains available. The tutor next assigns key exemplar resources such as images and video clips to pages in the computer assisted learning package. Finally the tutor sets the computer assisted learning package player in the local virtual learning environment to release the activity to their students at the start of PBL exercise. The computer assisted learning package becomes available to the students at the start of the PBL activity and they work through it individually but with shared chat/discussion facilities to afford discussion and collaborative problem solving. The virtual patient data appears inline in the computer assisted learning package player pages.
Transactions	Tutor adapts and merges virtual patients to create a new long-case virtual patient. Tutor creates computer assisted learning package and sets up sections within it delimited by key questions or decisions. Tutor assigns virtual patient data to different sections of the computer assisted learning and sets release to cumulative so that data, once released, remains available. Tutor adds media resources to computer assisted learning package. Tutor sets up computer assisted learning package in player in local virtual learning environment. Tutor schedules release of activity through the virtual learning environment. Students access computer assisted learning package player via local virtual learning environment. Students work through exercises.
Exceptions	Various alternative authoring and runtime steps and players. virtual patients are not merged but run as separate virtual patient instances.

Notes: a long case PBL activity is one where the student works through an extended patient case, often of a chronic condition encompassing many presentations, interventions etc. The nature of the computer assisted learning package is not outlined here as this could be achieved in many different ways; it is assumed here that virtual patient data is rendered inline in the computer assisted learning package player.

ID	MVP-WP-04
Title	Global State Model + Activities + VPDs + Media Resources + Data Availability Models
Players	'virtual ward' game engine, student, activity service, VPD service, resource service, Data Availability Model service
Assumptions	Suitable tools and resources exist, all users have access to suitable networked computers (or equivalent). In particular the virtual ward game engine is set up with appropriate activities and follows basic game interaction affordances (physical movement, acquisition and use of resources and tools etc).
Description	A student wants to practice her decision-making skills using the online 'Virtual Ward Round' provided by a national clinical education organisation. She logs on to the Virtual Ward website and downloads a copy of the virtual ward game engine player. She next creates a new activity profile assigning it a role, area of specialty, educational level and name. The student next starts the virtual ward game engine, which immediately presents her with a simple introductory orientation activity. The student works through the activity successfully and is next presented with a clinical case activity. The activity allows her to work through a patient presentation drawing upon clinical data (VPD) and media resources to form a diagnosis and treatment plan. The virtual ward game engine only allows her to proceed once she has successfully diagnosed the problem and presented a suitable treatment regime. Once the student has done this she is presented with a different and slightly harder case - and so on until all the cases for her educational level and specialties. Feedback (both environmental and critical) is given at every stage. Her performance metrics for each completed section are stored, thereby allowing the student to continue where she left off and also to work with several different profiles. Profiles in turn allow her to experiment with different educational levels and specialties.
Transactions	Student logs on to virtual ward website and downloads virtual ward game engine player. Student creates new profile and starts new session. Virtual ward game engine checks for completed sections and finding none presents training scenario. Student works through orientation activity. On completion of orientation, the first basic case activity (matching profile educational level and specialty is loaded), which in turn loads VPD and media resource data from their respective services. Student works through first case activity and receives feedback from the virtual ward game engine environment and by text or audio messages. Student's input and performance metrics are stored as part of the student's user profile.

Exceptions	Alternative game engines (see notes below). Loading an existing profile would allow the student to continue where she left off. Virtual ward game engine player runs inline to a web-browser so no download.

Notes: although described as being based on the heuristics of 'first person shooter' or 'first person puzzle' video games, this use case could also be instantiated in a collaborative mode or in a simple text environment based on principles of Multi-user Object Oriented Systems that could run on mobile devices like advanced mobile funs and networked PDAs.