Smart Contracts for e-Learning

Tsz Yiu Lam
1443964

Supervisor: Dr. Brijesh Dongol

Department of Computer Science
Brunel University London

A report submitted in partial fulfillment of the requirements for the degree of
BSc (Hons) Business Computing

Academic Year 2017 - 2018

I would like to dedicate this paper to Mum, Dad, Vivien, Viviana and Jorden.

Abstract

The properties of a blockchain could bring new features to e-Learning. Properties such as immutability and peer executed smart contracts could bring a new level of trust, transparency and personalisation to the education market.

This project focused on features that would increase transparency in assessments, and curriculum personalisation. They were identified as two of the key concerns in the current UK higher education industry that can be improved by a blockchain powered e-Learning platform.

The logic of the smart contracts and data models for such a platform were proposed. A working prototype was also developed based on the IBM Hyperledger Composer project.

Acknowledgements

The formatting of this report is done with Krishna Kumar's Cambridge University Engineering Department PhD thesis LaTeX template, and with referrence to a Microsoft Word template provided by Dr. Simon Kent.

The following open source software libraries and free resources were used to complete the demonstrator:

- Material Icons, from https://material.io/icons/
- Vue.js framework, from https://vuejs.org/
- Vue Material package, from https://vuematerial.io/
- vue-youtube-embed component, from https://github.com/kaorun343/vue-youtube-embed

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Tsz Yiu Lam

1443964

Academic Year 2017 - 2018

Table of contents

List of figures

List of tables

Chapter 1

Introduction

The global e-Learning industry already generates US$60 billion per year, and by 2019, over half of all courses will be taken online (Pantò and Comas-Quinn, 2013, p.17). This rising trend presents an opportunity to improve higher education.

Some current problems in higher education are related to transparency (more in Chapter 2.2.1). Tension exists between the educational provider and the learners over assessments. "There is abundant evidence that assessors are not particularly good at making exams valid, reliable, or transparent to students." (Brown Jr, 1999, p.62).

There is also a lack of curriculum personalisation for higher education learners in the UK [TODO: cite interviews, other refs]. Condie and Munro (2007) pointed out that the personalisation of the education curriculum for learners helps "overcome barriers, raising self-esteem and achievement".

Being able to conduct assessments and generate credentials in a transparent, trustworthy way would be central to a future e-Learning marketplace that is open, trusted and autonomous. This is where immediate value could be provided by blockchain systems and smart contracts.

A blockchain is a type of database that is spread across multiple sites, such as different institutions, companies or participants. A "block" of records is "chained" to the next with a cryptographic signature, creating permanent records through a consensus corroborated by all the operators (Walport, 2016, p.17). The verifiability of all actions on a blockchain gives systems an inherently high degree of integrity and transparency. Some types of blockchain can also have permissions that enables granular transparency and privacy (Walport, 2016, p.22), which fits well in the education paradigm where personal data can be highly sensitive.

Smart contracts are "contracts" that are "defined by the code and executed (or enforced) by the code" (Swan, 2015, p.16). They are self-executing code embedded in a blockchain that defines the

rules and penalties around an agreement and could automatically enforce those obligations (Gulhane, 2017), and can be used to exchange or transfer digital assets when certain conditions are met.

The potential of blockchain-enabled systems in education has been noted by the community, with Swan (2015, p.62) proposing that "learning smart contracts could automatically confirm the completion of learning modules through standardized online tests". Appropriate configurations in permissions and visibility can also provide improved security and privacy to e-Learning.

1.1 Aims and Objectives

The aim of the project is to design an e-Learning platform that fulfills educational assessments and rewards with smart contracts on a blockchain, providing improvements in assessments and curriculum personalisation for learners and teachers.

To satisfy this aim, the following objectives are planned:

1. Identify issues in education and e-Learning that can be improved by a blockchain-based system.
2. Develop data models and smart contracts in the proposed blockchain for e-Learning.
3. Develop permission models for the e-Learning blockchain that balances appropriate access and privacy protection.
4. Build a demonstrator system that includes client-side applications for learners and teachers.
5. Evaluate whether the blockchain-based demonstrator tackles the issues in education and e-Learning identified in objective 1

1.2 Project Approach

1. Review literature on current issues in e-Learning and education, and existing work in blockchain in education.
2. Further gather requirements for a blockchain solution for e-Learning using interviews with stakeholders.
3. Design data models, smart contracts and permission rules for assets and participants in the proposed blockchain solution.
4. Analyse popular blockchain development platforms that can be used to produce the desired solution.

5. Build the distributed ledger network and client side applications for learners and teachers.

6. Evaluate the design of the deliverables using interviews with stakeholders and relevant subject matter experts.

1.3 Dissertation Outline

Chapter 2 discusses the background for my project, and identifies some key techniques that can be adopted during the development of the proposed solution.

Chapter 3 explains how the project will be undertaken . . . etc, etc.

Chapter 4 requirements incl. primary data from interviews

Chapter 5 design

Chapter 6 implementation

Chapter 7 evaluation

Chapter 8 conclusion, future work

Chapter 2

Background

In this chapter we will take a deeper dive into the issues around education and e-learning that is relevant to this project, and

2.1 Literature Review in Education and e-Learning

Identifying issues in traditional higher education today that a blockchain-based system can better tackle is one of the objectives of this project. This informs the scope of the project and the design of the deliverables.

There is an abundant amount of pedagogy and learning method research, which focuses on the instruments and mode of delivery. These include methods such as "scaffolding", "constructivism", "problem-based learning", and "active learning" (Ali, 2005). However, this research area is considered out of the scope of discussion for this project, which does not aim to provide new insight into ICT-enhanced pedagogies. Instead, this project is interested in mapping components of e-learning, such as delivery, assessment, and record keeping in a more general-purposed, pedagogically neutral manner to a blockchain.

2.1.1 Assessments and Transparency

Assessment is arguably the most important process in the business of education as it "drives what is learnt and taught" and "converts learning into credentials". (Campbell, 2010, p.160)

Brown Jr (1999) summarises examples of popular sentiments learners held about both continuous assessments and traditional exams, such as:

1. Assessment tasks do not increase students' want to learn, only their need to learn, promoting unhappiness;

2. Invalid and unreliable marking due to speed or fatigue of assessors, plagiarism and unwanted collaborations, etc.;
3. Sub-optimal levels of feedback after many types of assessments;
4. Students feel forced into surface learning.
   (Brown Jr, 1999, p.62-65)

The importance of assessments, coupled with popular unhappiness and mistrust amongst learners towards them, increases the tension between the teacher (or educational provider) and the learners.

Suhre et al. (2013) looks into motivation on study progress in a higher education setting by collecting data from 168 first-year university students for six months. The study found three main factors that motivates academic progress: intrinsic abilities, personal motivations such as a need to achieve or fear of failure, and transparency in exams and assessments.

Transparency here refers to both the clarity of assessment goals and the procedures for assessing these goals. It should be clear to learners what knowledge is required for a sufficient level of mastery (Suhre et al., 2013). The difference this makes was significant:

- Students' perceptions of degree programme organisation and transparency of exams are significantly correlated with academic performance;
- Academic pressure is substantially influenced by the perceived transparency of assessments.

The earlier [p.100]Bryan and Clegg (2006) research echoed the same argument, that students realise their full potential when they understand the assessment task, marking criteria and expected standards.

An improvement in the transparency of goals, procedures, knowledge required of assessments and an increase in feedback can directly tackle some of the negative sentiments listed above from Brown Jr (1999), such as 1, 2 and 3.

2.1.2 Personalisation in Education

[TODO: Cover personalisation broadly and in terms of curriculum (which modules to take, customised passing thresholds) which can be negotiated on the blockchain. To be added if there is time for the project to cover this area.]

Personalisation is regarded as the solution to traditional bureaucratic state education that is irresponsive, inflexible, over-regulated, with the 'one size fits all' approach. The case for personalisation assumes that prior state provision was, if not actually totalitarian, certainly machine-like, crude, churning out standardised goods, insensitive to diversity. (Bragg, 2014)

Current research into personalisation (or customisation) of education is broad and covers both the personalisation of pedagogy and curriculum. Notably, research points to a growing appreciation of the need to support and encourage learner control over the whole/entire learning process (Dron, 2007).

Green et al. (2005) summarised four key areas pivotal to enabling personalised learning through digital technologies. The pedagogy should:

- ensure that learners are capable of making informed educational decisions;
- diversify and recognise different forms of skills and knowledge;
- create diverse learning environments; and
- include learner focused forms of feedback and assessment.

The design of a learning platform that aims at supporting personalisation must therefore focus on facilitating these four actions.

(p.7)

http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8527.2008.00411.x/full

2.1.3 e-Learning Systems

E-learning has been growing as an industry and research area, and the Learning Technology Standards Committee was set up within the IEEE Computer Society to devise relevant standards. In 1999, the Learning Technology Systems Architecture (LTSA) standard was published (See Figure 2.1) and was last revised in 2003.

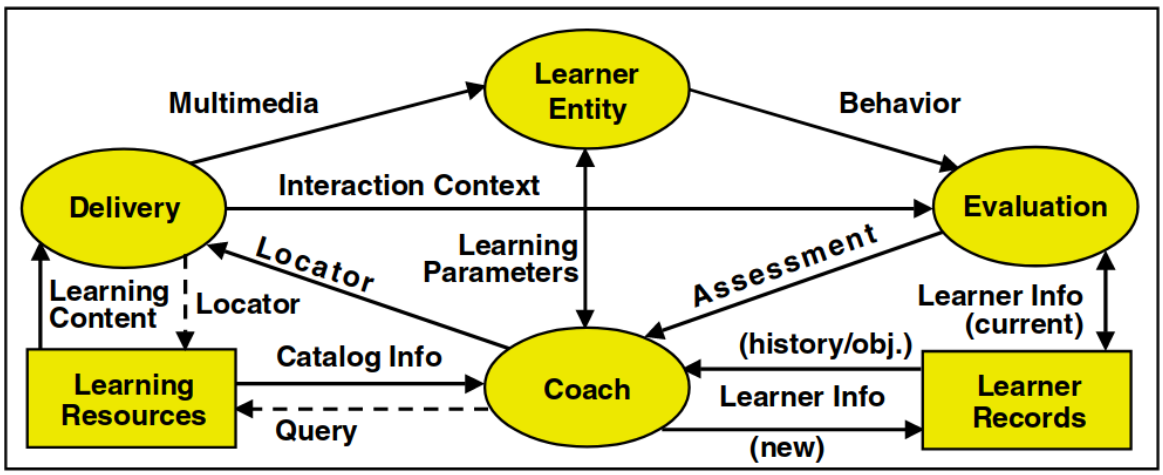

Fig. 2.1 Learning Technology Systems Architecture, IEEE Standard (IEEE Computer Society, 2013, p.9)

LTSA is "pedagogically neutral, content-neutral, culturally neutral, implementation-neutral, and platform-neutral"(IEEE Computer Society, 2013, p.1). It provides a valuable way of organising the

scope and discussion in this project. It identified four main components: learner entity, coach, delivery, evaluation; two main stores: learning resources, learner records; and the information flows or actions between them.

Identifying the properties of a blockchain-based system that could improve these components, stores and flows is critical to this project. For example, the distributed, immutable storage of learner records could provide extra security.

2.2 Security and Privacy

The security of e-learning systems have also been a concern. For example, El-Khatib et al. (2003) noted that “while many advances have been made in the mechanics of providing online instruction, the needs for privacy and security have to-date been largely ignored. At best they have been accommodated in an ad-hoc, patchwork fashion.”

The consequences of cybersecurity breaches have also become more and more expensive. For example, when the General Data Protection Regulation (GDPR) comes into effect across Europe in May 2018, the maximum fine for poor practices and data breaches will be £17 million or 4% of global turnover (Denham, 2017).

The scale and severity of historic breaches of of internet services has been worrying. Most notably in the e-learning industry, the education platform Edmodo was hacked and 77M account details were lost and on sale on the dark web, endangering students, teachers and parents who are account holders (Opsecmonkey, 2017).

The sizable threat and consequences makes a "security by design" and "privacy by design" approach for future e-Learning systems very important.

2.3 Properties of Blockchain Technologies

The advent of cryptocurrencies made blockchains an overnight darling, set to make significant disruptions in several industries such as financial services, currency exchanges, supply chain management, retail advertising and identity management (Bessonov, 2017).

The blockchain data structure is a timestamped list of blocks, which stores data about transactions that occur within the blockchain network. It only allows the insertion of transactions, not the update or deletion of existing transactions. Its ability to prevent tampering is known as "immutability". (Xu et al., 2016, p.182)

Blockchains can be classified into two types: one being a permissionless (public) blockchain which anyone can use and no central authority exist to allow or ban peers; the other a permissioned blockchain (can be public or private) where a central entity assigns read/ write rights to individual peers (Wüst and Gervais, 2017, p.1). Table 2.1 summarises the main differences in these two types of blockchain.

Table 2.1 Comparison of permissioned and permissionless blockchains, modified from Wüst and Gervais (2017, p.3)

Properties	Permissioned blockchains	Permissionless blockchains
Speed	Low throughput and slow latency	high throughput and medium latency
Peers	High number of both readers and writers	High number of readers, small group of writers
Consensus	Proof of work or proof of stake by miners	BFT protocols such as PBFT
Central Authority	No	Yes
Privacy	Can be achieved using cryptographic techniques but typically comes at the cost of lower efficiency	Reading rights can be restricted by central authority, readers and writers can also run separated parallel blockchains that are interconnected.
Verifiability	Observers can verify the state of the blockchain	
Redundancy	High, provided through replication across the peers	

Using blockchains as a data storage gives the system a very high degree of integrity. The public verifiability, redundancy (in Table 2.1) and immutability of the blockchain makes it very difficult to corrupt or lose the data stored.

2.3.1 Decision Framework for Blockchain Solutions

Wüst and Gervais (2017, p.3) proposed a decision flowchart (Figure 2.2) to determine whether a blockchain is the appropriate solution for a problem, and which type of blockchain is the most appropriate. Here is an analysis of our problem at hand with the decision flowchart steps:

1. **Do you need a store state?** Yes. Records in an e-learning system require secure storage.

2. **Are there multiple writers?** Yes. There are many different authorities, institutions, educators and learners involved in an e-learning blockchain that demands write assess into records.

3. **Can you use an always online Trusted Third Party (TTP)?** Wüst and Gervais (2017, p.2) described two options of using a TTP: delegate write operations completely to the TTP if it is

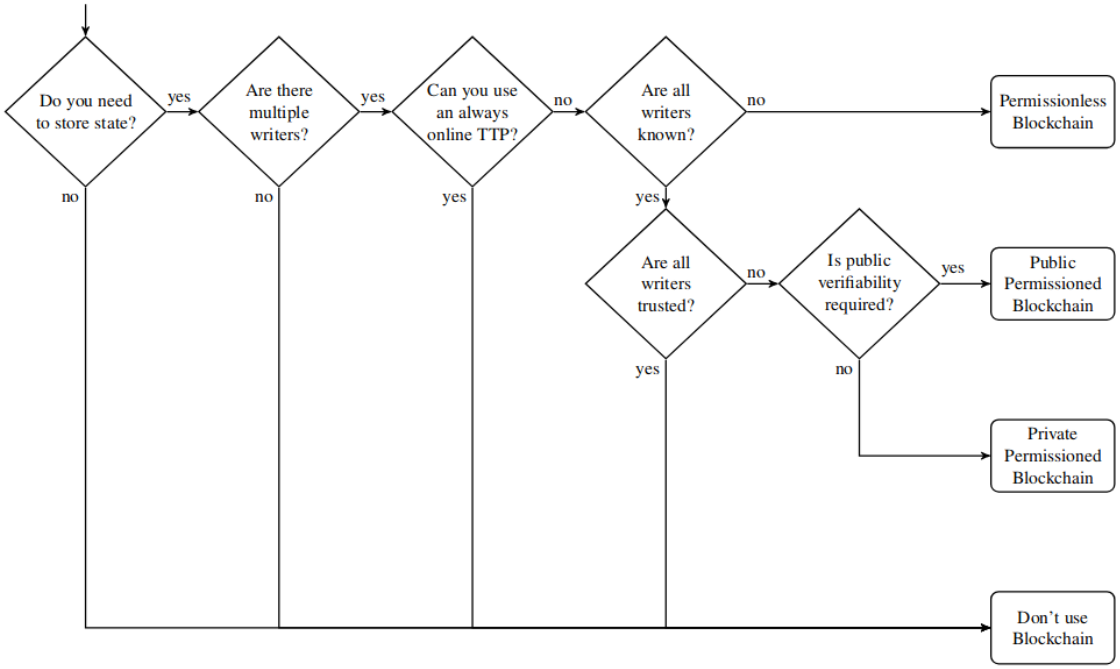

Fig. 2.2 "Do you need a blockchain?" flowchart (Wüst and Gervais, 2017, p.3)

always online so that it verifies all state transitions, or use the TTP as a certificate authority in the setting of a permissioned blockchain if the TTP is usually offline.

In the e-learning context, a TTP could be the e-learning platform provider. However, the delegation of write operations to the platform goes against modern principles of autonomy and independence for higher education institutions. Governmental education ministries in most modern states audits and regulates higher education without writing student records or conferring degrees. An always online TTP should not be used in order to replicate the real world context. This project will answer no for this step.

4. **Are all writers known?** Yes. Users of the system should be registered and not be anonymous to the system administrators. [TODO: Why? Are there arguments to anonymous education?]

5. **Are all writers trusted?** No. Malpractices from education institutions can occur, especially in the private, for-profit sector. In a future open e-learning market, it could also be possible for anyone to start offering education services. We cannot assume that all writers are trusted.

6. **Is public verifiability required?** Yes. One of the objectives of this project is to boost trust in e-learning credentials by increasing public verifiability of education journeys, increasing public accountability especially for stakeholders such as employers and postgraduate studies providers.

This process led to the recommendation of a **public permissioned blockchain** for this project.

2.3.2 Properties of Smart Contracts

Smart Contracts are self-executing code embedded in a blockchain that defines the rules and penalties around an agreement and could automatically enforce those obligations. They can effectively "cut out the middleman" to save time, and prevent disagreements about transactions (Gulhane, 2017). The term "chaincode" is also used interchangeably as a synonym for Smart Contracts (Valenta and Sandner, 2017, p.6).

There are three main properties of Smart Contracts (Swan, 2015, p.16):

- **Autonomous**: after launching and running, no further communication is required between a smart contract and its initiating agent;
- **Self-sufficient**: a smart contract should have the ability to keep itself alive when it needs to be, such as raising funds by providing services, and spending them on computing power or storage;
- **Decentralised**: a smart contract does not exist on a single server, they are distributed and self-executing across all of the blockchain peers.

These properties ensure effective operation of the logic defined. In an e-learning context, this can potentially be used to govern how teaching, evaluation and feedback take place, enhancing protection for the consumers/ learners.

2.4 Overview of Blockchain Development Toolkits

This project will involve the design of Smart Contracts for e-Learning transactions and building a demonstrator network and applications. A review of the popular blockchain implementations and development toolkits on the market is necessary. See Table 2.2 for a overview and below for more commentary.

Bitcoin is included in the Table only as a point of reference. Building with the bitcoin blockchain is not considered for this project because of its lack of support for Smart Contracts (any kind of embedded logic or programmes).

Ethereum

Ethereum is famous for its Turing-complete Smart Contracts capabilities, which allows entire decentralised applications (dApps) to run autonomously on its blockchain. It has a build-in cryptocurrency "Ether", which is used to reward miners that contribute to consensus, and to pay transaction fees. Developers of dApps can also issue their own currency inside their Smart Contracts.

Table 2.2 Comparison of key blockchain implementations, adapted from IBM Blockchain (2018) and Valenta and Sandner (2017)

	Bitcoin	**Ethereum**	**Hyperledger Fabric**	**R3 Corda**
Governance	Bitcoin developers	Ethereum developers	Linux Foundation	R3
Cryptocurrency	bitcoin	ether, and user-created cryptocurrencies	none, or user-created cryptocurrencies	none
Network	permissionless, public	permissionless, public or private	permissioned, private	permissioned, private
Transactions	anonymous	anonymous or private	public or confidential	
Consensus	proof of work	proof of work	PBFT	PBFT
Smart Contracts	none	yes (Solidity, Serpent, LLL)	yes (chaincode)	yes (chaincode)
Language	C++	Golang, C++, Python	Golang, Java, JavaScript	Kotlin, Java, legal prose

Valenta and Sandner (2017, p.3-4) noted that compared with Ethereum, which makes records accessible to all participants, permissioned blockchains such as Fabric and Corda provide "more fine-grained access control to records and thus enhance privacy". They also achieves higher performance due to a faster consensus mechanism that does not involve mining.

For this project, another crucial consideration against the adoption of the Ethereum environment is the lack of a central authority, which makes it impossible for the platform to kick unscrupulous actors off the blockchain.

Hyperledger Fabric

Valenta and Sandner (2017, p.7) described Fabric as a highly flexible "versatile toolbox". It has different roles for peers within the network, where they can act as clients (end users), peers (record keepers), endorsers (transaction verifiers), or orderers (transaction requester). The consensus mechanism is by default a Byzantine fault-tolerant (BFT) algorithm and can be customised. A cryptocurrency is not required, but could be developed with chaincode.

R3 Corda

Built following use cases in the financial industry, Corda notably augmented Smart Contracts with legal prose, making it a great tool for highly regulated environment. The development of a cryptocurrency is not intended or supported. (Valenta and Sandner, 2017)

Education is not a highly regulated environment in most countries and this unique selling point of Corda is not immediately attractive to this project.

2.5 Existing Efforts in Blockchain for Education

2.5.1 Blockcerts

Blockcerts is an open standard for blockchain certificates led by MIT's Media Lab. Education providers can use it to store the records of certifications they have awarded (See Figure 2.3). One bitcoin transaction is performed for every batch of certificates, with the certificates stored in the OP_RETURN transaction field on the bitcoin blockchain. This is paid for by the certificate issuer. (blockcerts.org, 2018)

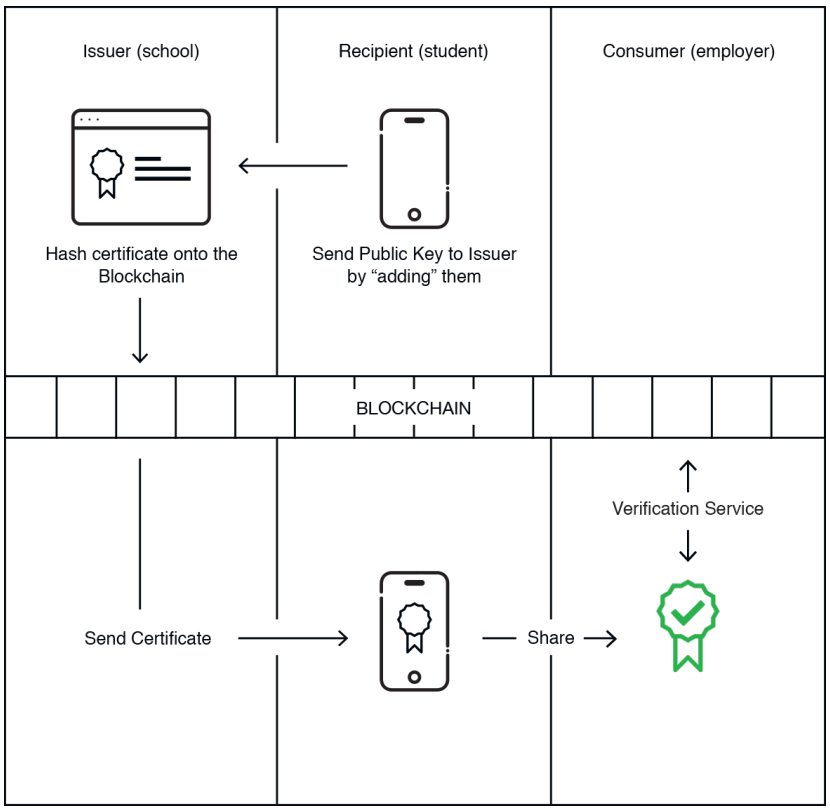

Fig. 2.3 How Blockcerts work (blockcerts.org, 2018)

2.5.2 Sony Global Education Blockchain

The Sony Global Education Blockchain is based on the Hyperledger project, an open source distributed ledger for businesses. It provides an application program interface (API) for developers at education institutes, allowing integration with third party applications. It aims to provide tamper-proof, secure storage of learning history data for institutions. (Sony Global Education, 2017)

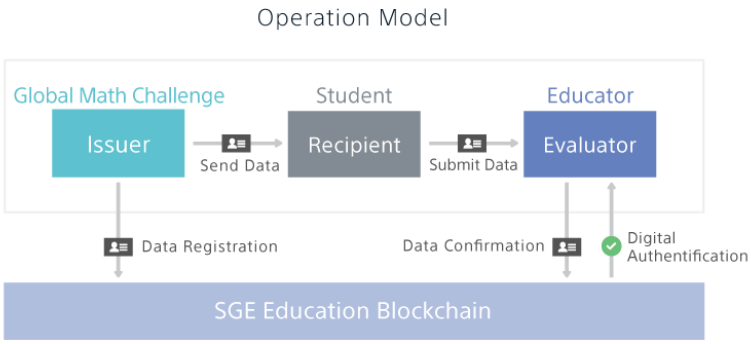

Fig. 2.4 Operation Example of the Sony Global Education Blockchain (Sony Global Education, 2017)

2.5.3 OpenLearn Blockchain

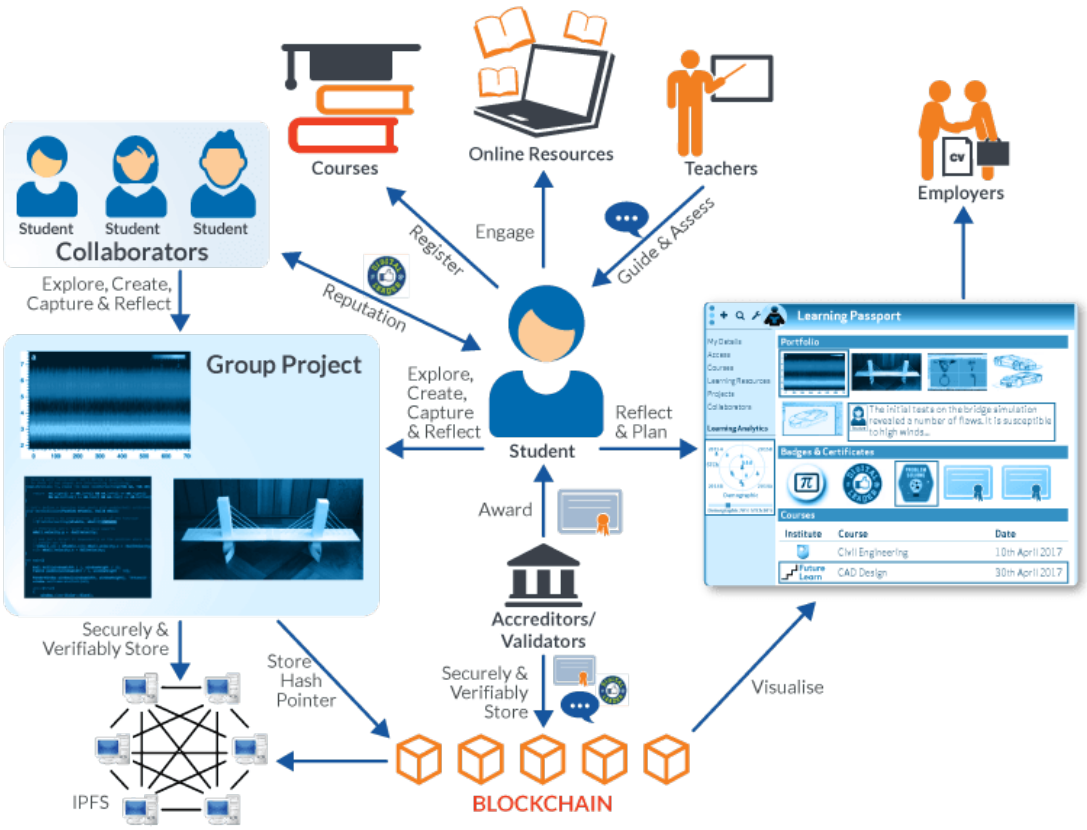

Fig. 2.5 A typical education scenario with the OpenLearn Blockchain (Open University, 2018)

The OpenLearn Blockchain project envisions the creation of blockchain based ePortfolios (See Figure 2.5). They are currently demonstrating this with an experimental plugin for Moodle, a popular course management system, with which achievement badges can be stored on the Ethereum blockchain. This system currently allows students to register for courses and receive badges which can be viewed in a student Learning Passport. The transactions are peer-to-peer: in principle no host institution is required for the awarding of accreditation. (Sharples and Domingue, 2016)

2.5.4 Novelty of the Proposed Project

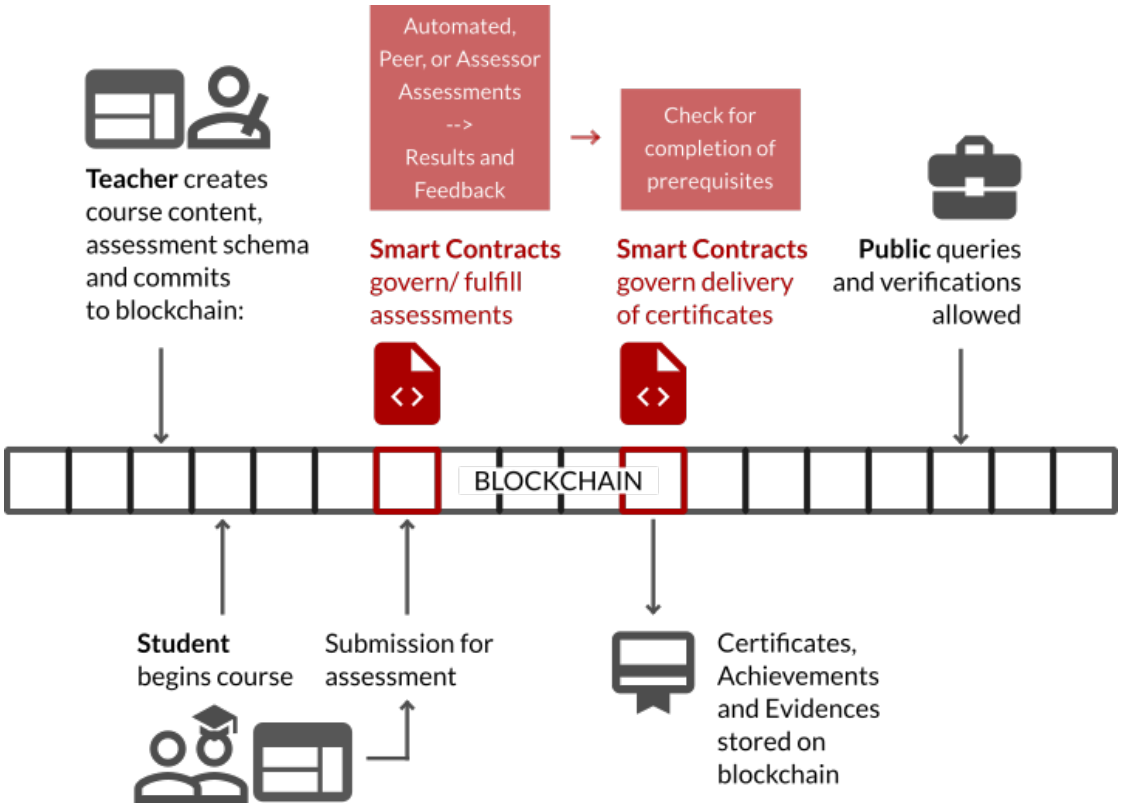

Fig. 2.6 Original diagram providing a high level view of how the project proposes automating assessments with Smart Contracts

All of the above three notable efforts focus on identity management and record keeping for education. This project will aim to extend these efforts by proposing Smart Contracts that automate assessments (See Figure 2.6) and deliver personalised curricula.

The vision of this project will also be to create an e-Learning marketplace that teachers can use directly, instead of a blockchain network that education providers will have to consume through APIs. This makes for a lower cost of entrance in terms of technological know-how and investment.

Summary

Combining the recommendation for a public permissioned blockchain from the framework provided by Wüst and Gervais (2017) (See Section 2.1.1), and the brief analysis above, Hyperledger Fabric stands out as the best platform which could allow for a lot of future work. Backed by the Linux Foundation, Hyperledger projects have extensive documentations and a large, active community.

The discussion following this chapter will now commit to using Hyperledger Fabric as the blockchain platform for this project.

[TODO: intro frameworks before existing work]

Chapter 3

Approach

3.1 Scope

The aim of this project focuses on improving assessments and curriculum personalisation, and does not require coverage of all elements in the Learning Technology Systems Architecture (LTSA, Chapter 2.2.1). See Figure 3.1 for which parts of LTSA this project will attempt to cover, and Table 3.1 for the mappings from LTSA elements to the blockchain design objects in this project.

Components such as delivery and the storage of learning resources will be out of the scope of this project. A blockchain is also not the ideal storage service for multimedia data such as learning content and delivery activities due to the inherent high latency in consensus. Security and redundancy of these data is also unnecessary.

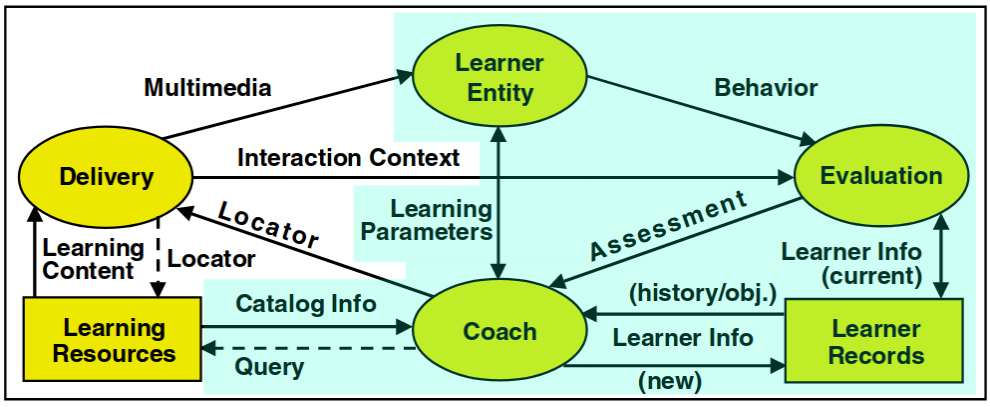

Fig. 3.1 The Learning Technology Systems Architecture (IEEE Computer Society, 2013), with components covered by this project highlighted in cyan

Table 3.1 Mappings between LTSA elements and the BlockChain objects in this project

LTSA Elements	BlockChain Objects
Learner Entity (component)	Learner
Coach (component)	Teacher
Evaluation (component)	Assessment Results, Feedback
Behavior (flow)	Submission
Catalog Info (flow)	Course Modules, Module Units
Assessment (flow)	Assessment (Automatic Assessment, Assessor Assessment, etc.)
Learner Info (flow)	Curriculum (a collection of Course Modules)
Learning Parameters (flow)	Negotiation
Learner Records (component)	Certificates, Submission, Assessment Results

3.2 Project Timeline

The project largely follows a linear process, with key steps in planning, requirements elicitation, implementation and evaluation shown in Figure 3.2.

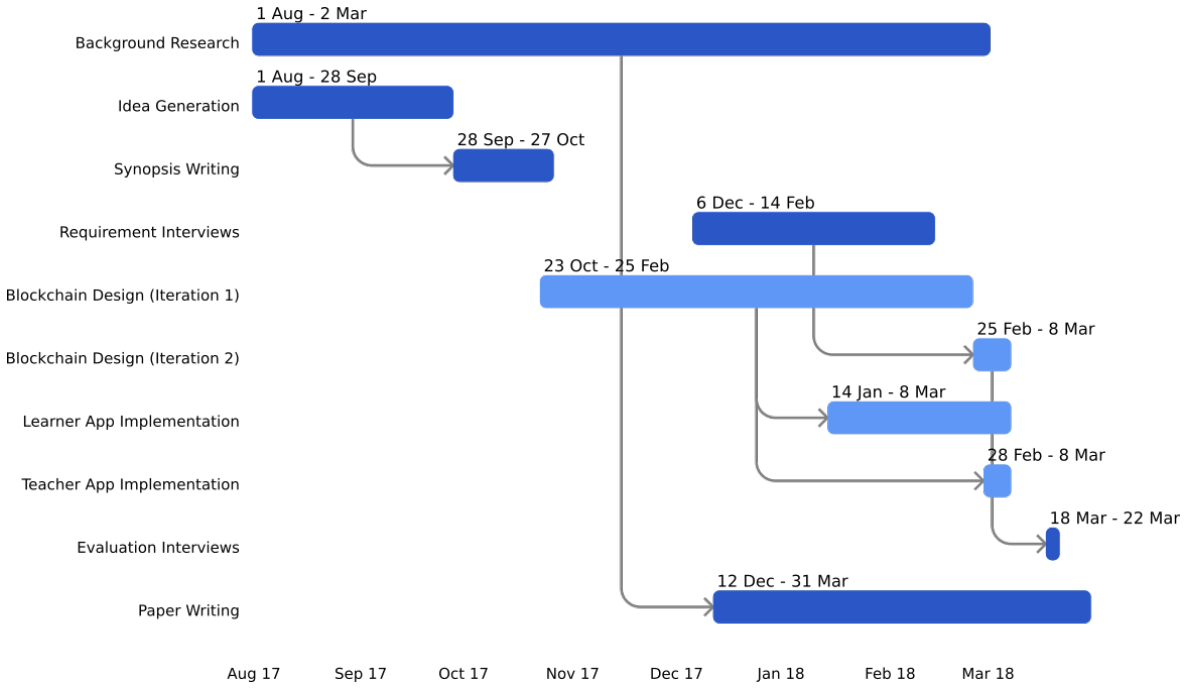

Fig. 3.2 A timeline of the stages of actions in this project and their dependencies if any

3.3 Lean Project Management

Lean as an approach to project management was first developed by Toyota in 1950s, later applied to software project management in the 1990s. Experiments have shown that it could:

- prevent workload from exceeding the capacity of the team;
- provide quick, high value, incremental deliverables;
- reduce technical and market risk;
- provide continuous improvements and reduce error rates.
  (Middleton and Joyce, 2012, p.30)

Two kanbans were used to manage the project processes: a Kanban for implementation and a Kanban for research and writing. The implementation Kanban used a loose adaptation of the development phase Kanban from Middleton and Joyce (2012, p.25), without ideation, user acceptance testing and release stages. See Table 3.2 and 3.3 for what the Kanban categories adopted are:

Table 3.2 Implementation Kanban used for the project

Minimum Marketable Function	Developing	Development Complete	Tested

Table 3.3 Research Kanban used for the project

Reading/Activity	Writing	Review	Completed

The Kanbans were kept on the online collaborative Kanban platform Trello. This is due to the good extensibility of Trello, with plugins allowing the attachment of links and documents from many other platforms.

3.4 Agile Software Development

There are two main agile software development principles that this project has chosen to adopt:

- Working software over comprehensive documentation;
- Responding to change over following a plan.
  (Beck et al., 2001)

Some clear limitations exist, for example, Beck et al. (2001)'s Agile Manifesto asks for daily collaboration between customer and developers, but contact with stakeholders for this project has only

happen twice at the requirements elicitation phase and the evaluation phase. Beck et al. (2001) also asks for teamwork and interactions over processes and tools, but this project is an individual endeavour.

At the end of the implementation phase of this project, two main iterations were completed.

Chapter 4

Requirements Elicitation

Background literature review in Chapter 2.1 and 2.2 has driven the direction of the project and provided many of the functional requirements. This chapter describes the further collection of primary data, and provides the list of formal user requirements.

4.1 Primary Data and Analysis

Collecting primary data from shareholders of higher education e-learning systems would be able to:

- reaffirm and further develop requirements obtained from literature
- obtain further requirements from real world painpoints and goals

4.1.1 Methodology

[TODO]

Method: qualitative

Instrument: Why interviews?

Sample: Who are they? Why? Convenience Sampling

BREO

A total of five interviews were conducted between December 2017 and February 2018: two with education professionals and three with student representatives. See table 4.1 for a more detailed description of these participants.

Table 4.1 Participants in primary data collection interviews

Participant	Characterisation
Educator A	lecturer in higher education for over 20 years, and an experienced higher education administrator
Educator B	lecturer in higher education for over 20 years, with research interests in e-learning interactions, effectiveness and acceptance
Student C	a university course representative for 3 years, which involves collecting and communicating student feedback and attending staff-student liaison meetings
Student D	a university peer assisted learning leader for 2 years, helping out lower level students with their academic work by facilitating peer discussions, and escalating common problems to academic staff in debrief sessions
Student E	a university course representative for 2 years and a peer assisted learning leader for 1 year

4.1.2 Interview Results and Analysis

The raw data from interviews (transcripts) were contextually analysed and grouped into problem statements (PS), these problem statements were then sorted into groups to produce an affinity diagram (See figure 4.1).

Below you will find the sample questions asked for each group, detailed descriptions and relevant transcript snippets for each problem statements. Any specifics regarding course, staff and event details have been anonymised.

On Assessments

Sample Question: What are the problems with assessments in higher education today? / Is there tension between students and teachers over assessments, and why?

PS1: Poor communication of assessment expectations

Four participants confirmed that the problem with transparency in expectations for assessments (as described in Chapter 2.1.1) does exist.

Educator B: "Often it is not clear to the students what they have to do... staff [should be] making sure that it is clear what is expected of an assessment."

Student C: "Sometimes [assessments] are unfair because they are based on things you have not learnt in lectures but have to read around... so not knowing what is actually assessed..."

Student D: "Students have a lot of problem understanding what to do in assessments, it is not clear enough how you can prepare for it."

Student E: "[Teachers] did not put it out clearly... all of the students where just lost on what they are

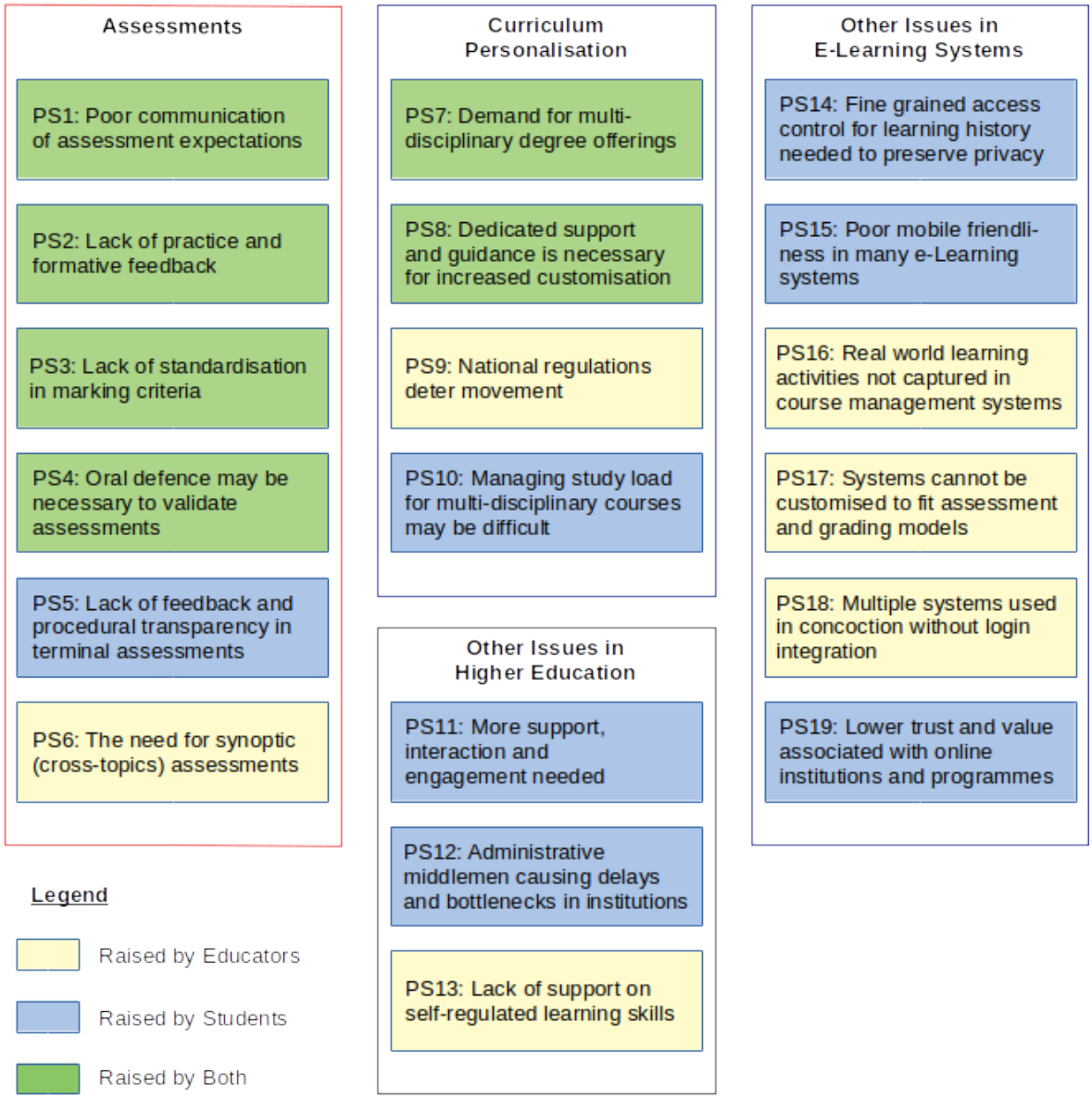

Fig. 4.1 Affinity diagram for problem statements devised from primary data

expecting from our coursework... people feel too stupid to ask those questions."

PS2: Lack of practice and formative feedback

Students transitioning from school to higher education would experience a drop in formative assessments (that do not affect their final grade) such as homework practices.

Educator B: "[Students] are not used to not practising what they have to do over and over again before [assessments]... and that's what they had done at school... and the students that I feel most need formative feedback also tends not to do them."

Student D: "[Students] really like past papers... Practice really helps... they are the best way of understanding what's coming in the exams."

PS3: Lack of standardisation in marking criteria and lack of automation in module review

Educator B: "In [our department] we have those assessment forms that we have to fill in, which ensures we at least but something for each of the [marking criteria] boxes... the marking scheme is sent to the module reviewer, who reads them and ensures they are clear. This is all manual... We are one of the few departments in the university that has anything like this. In lot of departments, these are not clearly defined by a long stretch."

Student C: "Having the marking criteria set in stone is actually the most important thing to students."

Student E: "[Assessments] are kind of fair, but it depends on who your tutor is... some tutors were giving As out way too easily... there should have been some sort of structure that the tutors followed."

PS4: Oral defence may be necessary to validate assessments

Educator A: "[Automated assessment] is problematic... to verify that is the viva (oral defence). When I talk to my group and ask what they got from that [automated] test they said they got 100... but when I ask them to explain it to me, they could not. So the final piece of validation that is needed before they move on from the level is to be able to question them on their answers. You can trust what's in the system but whether you trust the value of what's in a system depends on the viva."

Student E: "Some people that I know faked a whole coursework yet still got a banging grade even though they did not produce any [of their work]... no one is checking properly."

PS5: Lack of feedback and procedural transparency in terminal assessments

Student C: "One of the biggest issues is when you come out of the exams and you find out just the grade but you don't get to see the paper."

PS6: The need for synoptic (cross-topics) assessments

Educator B: "Assessments are very compartmentalised into their modules... that every module has its own assessments. This leads to students thinking that their learning in one module is irrelevant to another, whereas in the real world we draw knowledge from different areas."

<u>On Curriculum Personalisation</u>

Sample Question: What do you think are the current road blocks to offering more multi-disciplinary, personalised, or even multi-institutional arrangements for higher education?

PS7: There is a demand for multi-disciplinary degree offerings but only a few universities are capable of offering them

It was argued by *Educator B* that having the freedom to choose is better than universities defining programmes that encompasses two particular fields, because these defined degrees could have a low

intake and not be economically viable. However, universities today seem to struggle with bureaucracy with students who wish to choose modules outside of their programme.

Educator B: "Yes I do think there is a demand [for more multi-disciplinary offerings] and I know a lot of models that do work like this. The Open University and Oxford Brookes University do these style of degrees where students have some core modules that they have to do... surrounding that they can choose modules that then add up to a certain number of credits."

Student C: "Definitely if there is a bit more combination, or an allowance in terms of picking [modules] for your degree... I think there is definitely a demand."

Student D: "There is a range of students that you have to try and please... I did hear one complaint... they didn't understand how [one module] has anything to do with the course, that it wasn't that hands-on and practical. I [thought] it was really good personally."

PS8: Dedicated support and guidance is necessary for increased customisation

There are careers that require multi-disciplinary backgrounds, but there is a risk of students not making informed choices.

Educator B: "I think students need to be carefully guided through their choices and have good career advice... For example, the pharmaceutical industry is having a hard time finding people with a background in both IT and biology."

Student E: "There is no point if it is like doing two half degrees and they are missing out."

PS9: National regulations requiring programme outcomes and specifications deter movement

The UK higher education academic infrastructure requires all degree programme to lay out programme outcomes and programme specifications, which makes movement difficult.

Educator A: "If you want to move from [one institution] to somewhere else and you have 240 credits, the first thing I am going to ask you is, have you read my level 1 and level 2 outcomes? Are you in a place where you can, if you do level 3, meet the outcomes of the programme? ... [Our] credit model is not the same as the North American one, so actually movement between [institutions or programmes], despite the government always saying they want this, is mitigated against by the way we conceive education, which is about programmes representing... what you can do when you leave a programme."

PS10: Managing study load for multi-disciplinary courses may be difficult

Student E: "I don't think we have the time [to study many disciplines at once]... unless we have fewer modules... so their credits should be the same, they shouldn't take more credits because they chose to do another course."

Other Issues in Higher Education

Sample Questions: What else needs improving in higher education in general?

PS11: More support, interaction and engagement needed

Student D: "The way [teachers] interact sometimes is not that engaging... [they] could be giving out facts and information but they are not telling you... the interaction needs to be there... a lot of students would just come out [thinking] I don't understand what's just been said."

Student D: "The contact time students and tutors have is not always as good as we think it is... face to face contact, that's what students are not getting enough of... the support you get can really change the outcome of assessments."

Student E: "Some students need a bit more help than others. Some people are embarrassed to face their learning difficulties."

PS12: Administrative middlemen causing delays and bottlenecks in institutions

Student E has complained that feedback was not given on time to students, and the excuse given was that the teaching staff has sent the feedback to administrative staff but it takes time for them to upload it onto respective systems.

PS13: Lack of support on self-regulated learning skills

Educator B: "[Students] don't realise the difference between school and university... the way they will be expected to learn... they have to do a lot of self-motivated learning and a lot of them find that a huge challenge. There is not enough help and guidance for students to make that transition."

Other Issues in E-Learning Systems

Sample Questions: What other features would you like to see in a future e-Learning system for higher education?

PS14: Fine grained access control for learning history needed to preserve privacy

Student D: "I don't think it's a good idea for all the records to be available to the public, not just anyone... I think students are very private... perhaps [the student could] allow employees and other institutions to view it."

Student E expressed concern over aggregated data, such as gender comparisons and class rankings, arguing that institutions do not have rights to disclose that data without permission.

PS15: Poor mobile friendliness in many e-Learning systems

Student C has pointed out that any new system should be built to be responsive and mobile friendly.

PS16: Real world learning activities not captured in course management systems

Educator A: "There will be things that were part of the education experience that are outside of the system such as face to face contact."

PS17: Systems cannot be customised to fit assessment and grading models

Many of the assessment features and functions on e-learning systems are incompatible with institutional requirements, or even national requirements. *Educator A* pointed out how there is no global standard in grade point averages, and gave examples of staff:

- taking assessments outside the system, then putting the results back in
- mapping grades from a system using North American points (back to British grades)
- incorporating an extra viva (oral defence) assessment step outside of e-learning systems
- modifying a Scandinavian system that requires double marking for all assessments to making double marking optional

PS18: Multiple systems used in concoction without login integration

Educator B: "[Teachers] have to log into all of them separately... which I find frustrating. Students find it frustrating as well... Blackboard Learn [a course management system] is one set of software, their intranet is another, Wiseflow [an e-Assessment system] is another, another login system entirely for e-Vision [a student record system]. Usability becomes a major pain point."

PS19: Lower trust and value associated with online institutions and programmes

Student C: "People tend to think online learning is a lot easier and it is devalued straight away. When you come to a campus to do it... it would probably be the same process as an online course but it would have more value because of the institution having a physical existence and not just a digital one."

4.2 Formal Requirements

Table 4.2 lists out the functional requirements (FR) adopted by this project going forward. They are related to one or more of the problem statements (PS) gathered above from primary data, or to the literature reviewed in Chapter 2.

Table 4.3 lists the non-functional requirements (NR) adopted by this project going forward. They are related to one or more of the problem statements (PS) gathered, or to usability heuristics.

These requirements have been ranked with the MoSCoW prioritisation framework, which specified four levels of priority: Must Have, Should Have, Could Have, and Won't Have this time (Agile Business Consortium, 2018). The MoSCoW levels are given from mainly a system engineering

perspective in planning the minimum viable product for the demonstrator of this project, and do not necessarily reflect the priorities of the stakeholders.

Table 4.2 Prioritised list of functional requirements for this project

	Requirement Statements	Related To	MoSCoW
FR1	The system would store learner records on a blockchain	Ch 2.2.1 (LTSA)	Must Have
FR2	Teachers would be able to create and edit learning resources	Ch 2.2.1 (LTSA)	Must Have
FR3	Teachers would be able to create and edit assessments	Ch 2.2.1 (LTSA)	Must Have
FR4	The system would enforce the provision of learning outcomes, knowledge required and assessment goals at the creation or update of modules	Ch 2.1.1 (Transparency), PS1	Must Have
FR5	The system would enforce predefined assessments rules (eg. marking schemes, transparent procedures and feedback) with smart contracts	Ch 2.1.1 (Transparency), PS3, PS5	Must Have
FR6	The system would allow teachers to configure multiple assessments and formative assessments for modules	PS2	Could Have
FR7	The system would require vivas (oral defences) after automated assessments	PS4	Could Have
FR8	The system would provide flexible configurability for grade schema	PS17	Could Have
FR9	Teachers would be able to create a customised list of courses for a student, customising programme outcomes and specifications	Ch 2.1.2 (Personalisation), PS7, PS9	Must Have
FR10	The system should feature dedicated support channels between students, teachers and other advisors	PS8, PS11	Should Have
FR11	Students would be able to add assessment submissions on the blockchain	Figure 2.6 (Concept)	Must Have
FR12	The system would be able to generate certificates on the blockchain when a course has been completed	Figure 2.6 (Concept)	Must Have
FR13	The system would allow students to control access to their education history on the blockchain	PS14	Should Have
FR14	The system would provide one login for content delivery, assessment and record keeping	PS18	Should Have
	Requirements targetting PS6, PS10, PS13, PS16		Won't Have

Table 4.3 Prioritised list of non-functional requirements for this project

	Requirement Statements	Related To	MoSCoW
NR1	The client applications would have the same functionalities across devices and a responsive interface	PS15	Should Have
NR2	The client applications would fail safely and is error-forgiving towards the user		Should Have
NR3	The client applications would notify users of relevant events on the blockchain network		Should Have
NR4	The system should be able to reduce administrative work	PS12	Should Have
NR5	The system should be able to create more trust in online education providers and programmes	PS19	Should Have

Chapter 5

Design

Following the discussion in Chapter 2.3.1 and 2.5, the designs for this project will be created on top of Hyperledger Fabric, a public permissioned blockchain. It has become clear that a blockchain specific, high-fidelity prototyping tool is needed to create system designs such as data models and transactions.

5.1 Design Tool

Hyperledger Composer is an open source development toolset and framework that aims to accelerate time to value for blockchain projects. It offers business-centric abstractions, allowing business owners and developers to rapidly develop use cases and model a blockchain network. The design tools offered take the forms of:

- An object-oriented modelling language (.cto file) to define data models in the blockchain network
- JavaScript functions (.js file) to define logic for smart contracts triggered by transactions
- An access control language (.acl file) to define access rules for records on the blockchain (The Linux Foundation, 2018)

See Figure 5.1 for a visual explainer of how Hyperledger Composer helps designers and developers create these high level definitions.

A significant advantage of using Hyperledger Composer is its ability to package these prototype definitions and deploy it to Hyperledger Fabric, our target blockchain platform. This will speed up the implementation of the proposed demonstrator applications in the next stage.

Throughout the design process, the Hyperledger Composer notations are converted or drawn into UML sequence diagrams, class diagrams and flowcharts. PlantUML, an open source language-to-diagram drawing tool, and TikZ, a LaTeX package that creates graphic elements, were used.

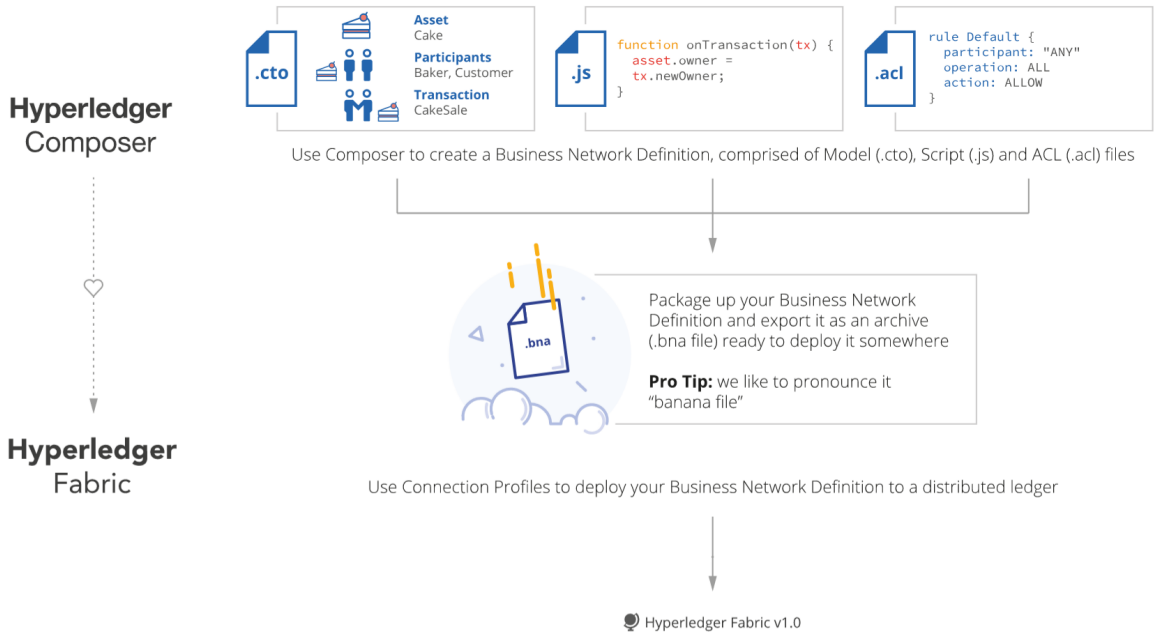

Fig. 5.1 Components in the Hyperledger Composer framework and how it deploys to Hyperledger Fabric (Cuicapuza, 2017)

The discussion below may regularly refer back to the functional requirements (FR) and non-functional requirements (NR) defined in the previous Chapter 4.

5.2 Transaction Sequences

A transaction is the only activity that a peer can perform to alter the state of a blockchain. Designing the sequences of transactions necessary to fulfil the desired user journeys will be able to provide a good overview of the work ahead. It will also shine a light on what data objects and properties must be defined. Two overarching use cases are considered: assessment and curriculum personalisation.

5.2.1 Assessment Use Case

These four transactions are required to complete the assessment use case (see Figure 5.2):

- CreateModule: a transaction ordered by a teacher to store metadata about a course module, its units and assessments onto the blockchain;
- AddSubmission: a transaction ordered by a learner to store a submission (assessment attempt) on the blockchain, this could return the result of the assessment if the result is returned by an automatic (machine) marking service;

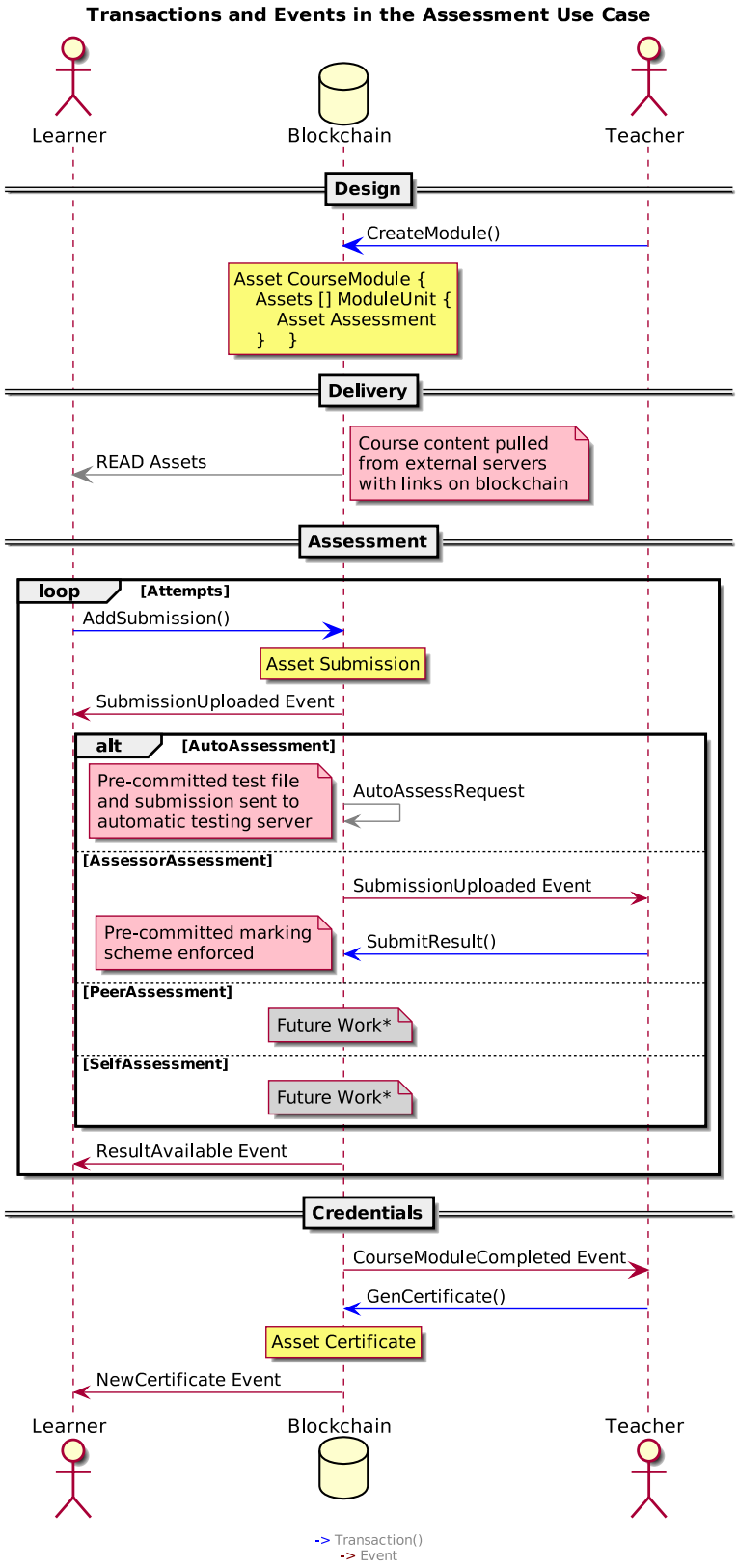

Fig. 5.2 A UML sequence diagram denoting the assets, transactions and events between Learner and Teacher participants on the blockchain for the assessment use case

- SubmitResult: a transaction ordered by a teacher to store details of an assessor assessment for a submission on the blockchain;

- GenCertificate: a transaction ordered by a teacher to create a new certificate on the blockchain.

5.2.2 Curriculum Personalisation Use Case

Similarly, we looked at the transactions required to build a minimum viable product that facilitates curriculum personalisation. A curriculum here is simply a list of course modules attached to a learner and a teacher (a personal tutor for the learner).

- ProposeCurriculum: a transaction ordered by a learner or a teacher to propose a new curriculum, or to proposed edits to an existing curriculum on the blockchain.
- ApproveCurriculum: a transaction ordered by a teacher to enrol a learner to the course modules in the learner's curriculum.

See Figure 5.3 for where these two transactions occur in a sequence diagram.

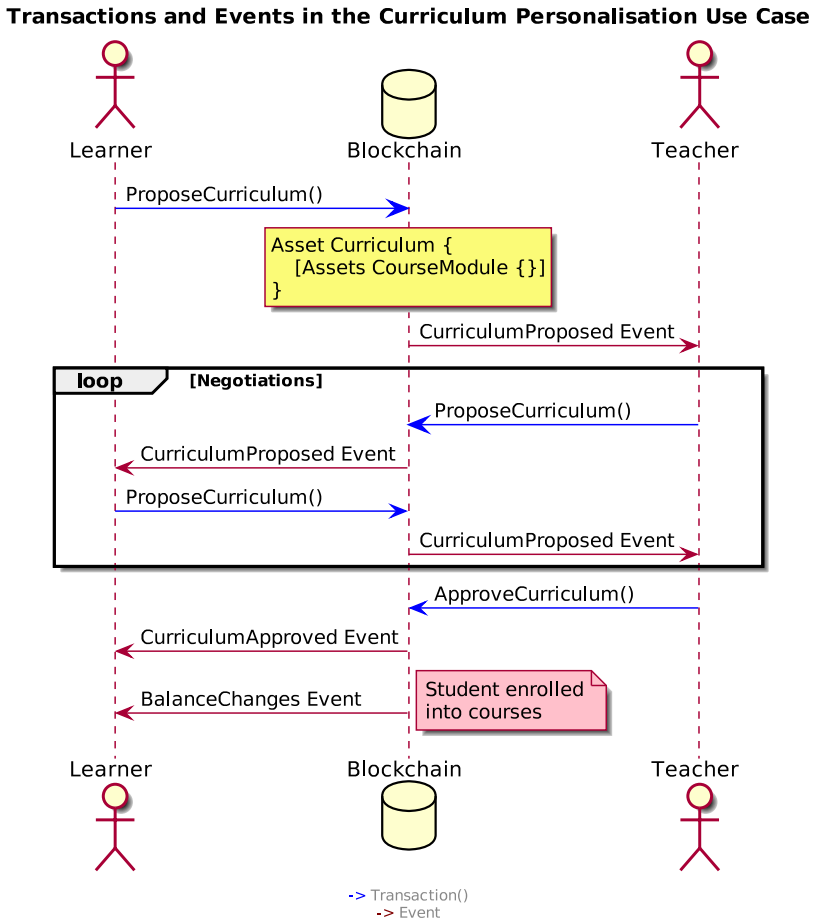

Fig. 5.3 Sequence diagram denoting the assets, transactions and events between Learner and Teacher participants on the blockchain for the curriculum personalisation use case

5.3 Data Models

Building a blockchain network requires a network-wide schema of what records are allowed to be created, updated and read. The Hyperledger Composer framework calls these resource definitions, and recommends defining objects inheriting three basic types: Participants, Assets and Concepts in its object-oriented modelling language (The Linux Foundation, 2018).

5.3.1 Participants

A participant is an actor in a blockchain network. A participant can create and exchange other assets by submitting transactions (The Linux Foundation, 2018). The network design for this project will allow the creation of three main types of participants:

- *Teacher*, which can be lecturers, teaching assistants, tutors, etc.
- *Learner*, which can be campus students, distant learners, etc.
- *Reader*, members of the public who are interested in querying or verifying records, such as employers and further education providers.

All three types of participants inherit an abstract (cannot be created) class *User*. The *nid* field that all *Users* must have would contain a one-way hash of their national identification number, which can be a driver's license number, social security number, etc. A hash is a form of cryptographical representation of data that is non-invertible. This allows the system to ensure that all writers in the system is known and unique, while protecting their privacy to the general public.

See Figure 5.4 for the detailed entity properties of all participant types in a class diagram.

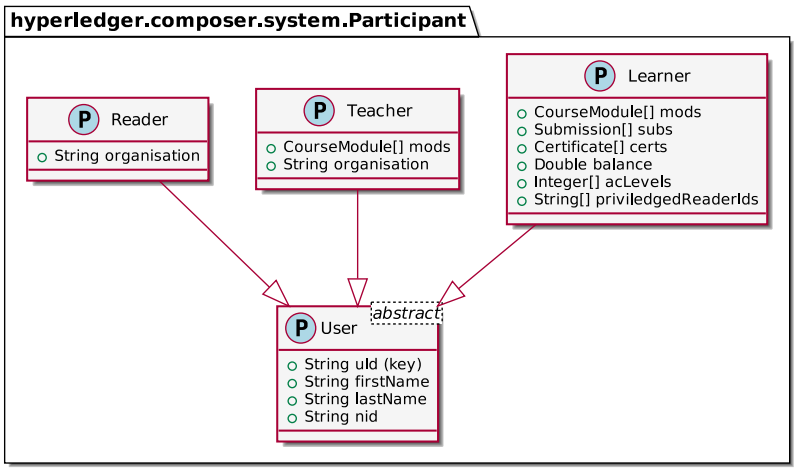

Fig. 5.4 A UML class diagram describing the participants defined on the blockchain

A notable design consideration is the mechanism for allowing tiered access control of learner information. The *acLevels* and *priviledgedReaderIds* fields in *Learner* store access control settings for two tiers of *Readers*, normal and privileged. This will be covered in more detail in an upcoming section.

5.3.2 Assets

Assets are "tangible or intangible goods, services, or property, and are stored in registries" (The Linux Foundation, 2018).

See Figure 5.5 for the detailed entity properties and relationships of these assets in a class diagram. The asset definitions are modelled carefully according to literature and user requirements. The special design considerations included:

- Types of assessments: A review of e-Learning and course management systems showed that online assessments could be grouped in four categories: self assessment, computer assessment, tutor assessment, and peer assessment (Paulsen, 2004, p.68). In this initial design, two of these types are considered: *AutoAssessment* (computer assessment), and *AssessorAssessment* (tutor assessment). They are daughter classes of the abstract *Assessment*.
- Transparency of assessment goals: the model contains mandatory fields to improve transparency in assessments, as identified by Suhre et al. (2013)'s research reviewed in Chapter 2.1.1. This includes the *knowledgeRequired* field in *Assessment* and *learningObjectives* in *CourseModule*, which encourage teachers to provide clarity over assessment goals.
- Transparency of assessment procedures: *Assessment* assets are on the blockchain and visible to all participants, improving transparency of procedures. They include the *gradeDescriptors* and *criteria* fields that which encourage teachers to provide clarity of assessment criteria, and give Smart Contracts the capabilities to enforce these criteria.
- Administrative Work for Personalisation: Primary data suggested that administrative and regulatory work required for curriculum personalisation could be daunting for institutions. The *programmeOutcomes* field in the *Curriculum* asset is one of the suggested regulatory steps for the UK market (see Ch4.X TODO). Making these administrative data available on the blockchain could allow future Smart Contracts to automate approvals and other administrative steps. This has the potential to reduce bureaucracy by eliminating the middleman.
- Content flexibility: It is anticipated that many markets, institutions, teams and teachers will have their own requirements, formats and templates for what their e-Learning content should

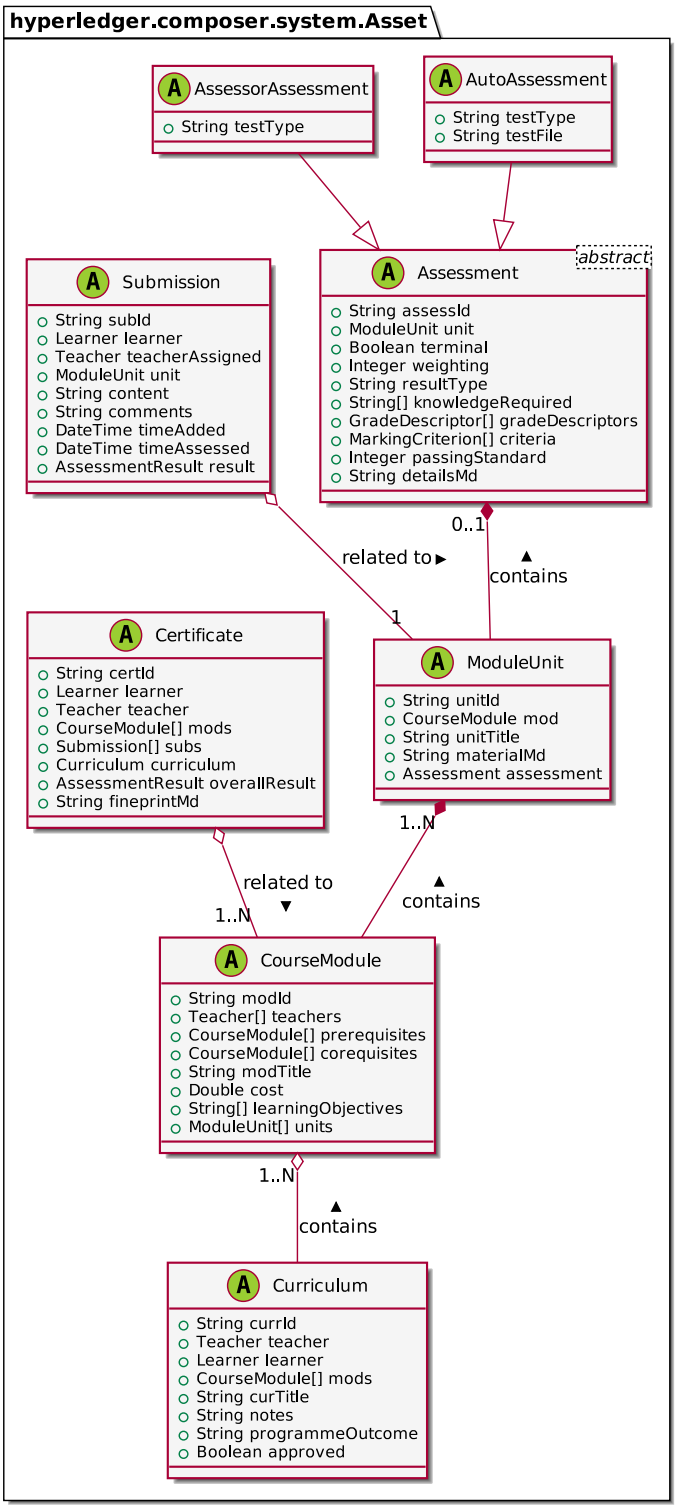

Fig. 5.5 A UML class diagram describing the assets defined on the blockchain

look like. To cater to this need for content and layout flexibility the blockchain will accept markdown syntax as input in fields such as *detailsMd* in *Assessment*, *materialMd* in *ModuleUnit* and *fineprintMd* in *Certificate*. Markdown is a popular text-to-HTML conversion tool for web writers (Gruber, 2004), with many internet forums and services releasing their own standards.

5.3.3 Concepts

Concepts are abstract classes that are not assets or participants. They are used to define custom properties contained by an asset or participant.

For this project: five of these Concepts were designed. These are related to modelling the assessment results, grade descriptors and criteria. See figure 5.6 for their detailed entity properties.

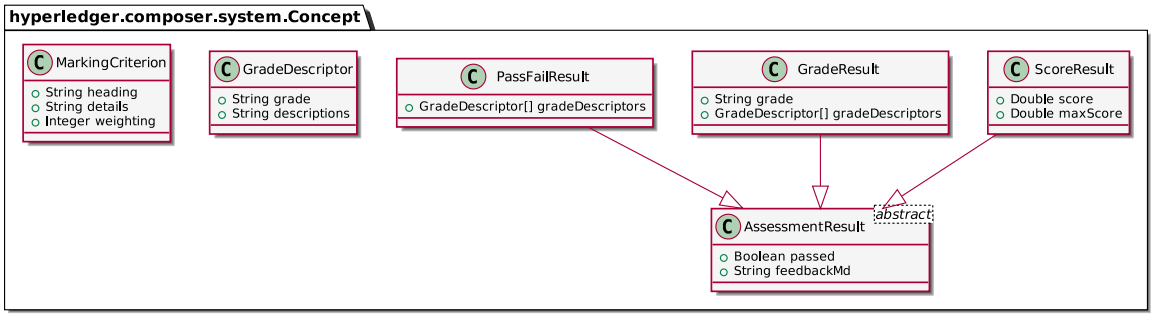

Fig. 5.6 A UML class diagram describing the concepts (other abstract classes that are contained as a field) defined on the blockchain

A matrix/ grid of assessment criteria against grade definitions, a common format used in schools (Bryan and Clegg, 2006, p.102), is used to record grade breakdowns on the blockchain. The *MarkingCriterion* and *GradeDescriptors* fields are designed to fulfil this feature.

AssessmentResult stores the overall result of an assessment or course module, as an example of how flexible this can be (TODO link to requirement), three example result formats are designed: *PassFailResult*, *GradeResult*, and *ScoreResult*.

5.4 Smart Contracts: Transaction Logic and Events

We have previously discussed the nature of Smart Contracts in Chapter 2.3.2. They are autonomous, self-sufficient and decentralised code that runs on a blockchain.

In Hyperledger Composer, Smart Contracts are called chaincode. They are run synchronously with a transaction order and a final output is required, to either accept the transaction, or reject it. Smart Contract chaincode can emit network-wide Events. These Events could be consumed by participants or applications.

The discussion below describes the six transactions previously planned in Figure 5.2 and 5.3, and the chaincode logic triggered by them.

5.4.1 The CreateModule Transaction

The transaction for creating a course module was kept simple, as this project is more interested in the assessment experience than the content creation experience. A *CourseModule* asset is uploaded with all of its *ModuleUnits* and *Assessments*.

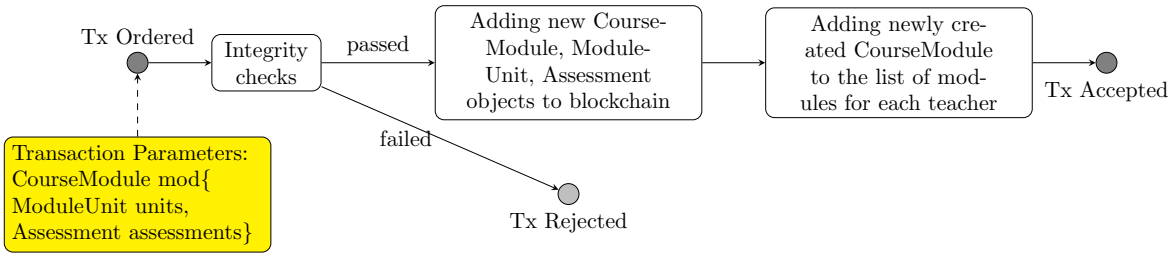

Fig. 5.7 Flowchart representation of the Smart Contract for the CreateModule Transaction (Tx)

Transactions that allow content editing and course archiving (making courses unavailable to new students) will have to be designed for a fully-fledged real world system, but we will defer them to future work for this project.

5.4.2 The AddSubmission Transaction

This transaction is ordered by a student to add a submission for the assessment of a module unit. The submission files will be compressed and converted into base64 data strings and attached to the *content* parameter.

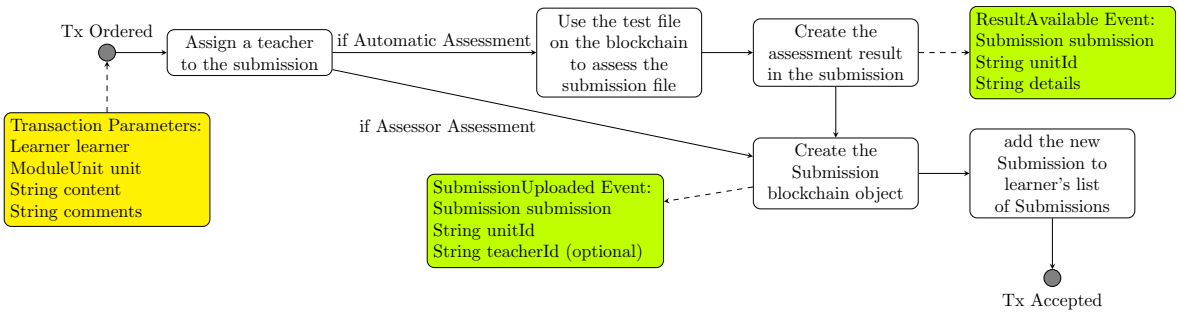

Fig. 5.8 Flowchart representation of the Smart Contract for the AddSubmission Transaction (Tx)

5.4.3 The SubmitResult Transaction

The assessor submits markings on a marking criteria against grade descriptor grid. This is uploaded as the transaction parameter *marks*, a one dimensional array where the index correlates to the marking criteria number and the value correlates to the grade number.

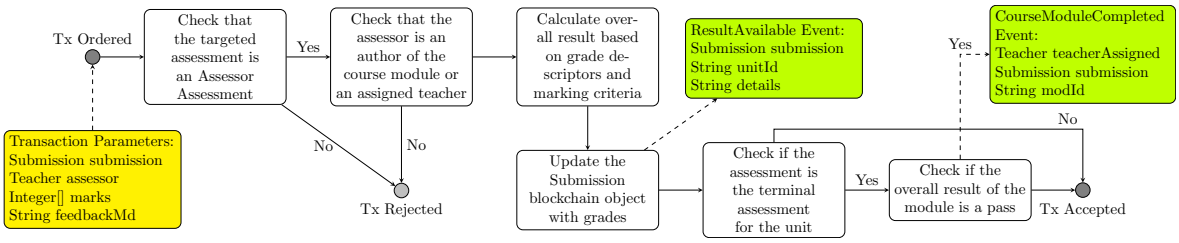

Fig. 5.9 Flowchart representation of the Smart Contract for the SubmitResult Transaction (Tx)

5.4.4 The GenCertificate Transaction

This transaction is designed to allow and encourage manual diligence over course completion evidences before issuing a certificate. It could also require multiple signatures for a certificate.

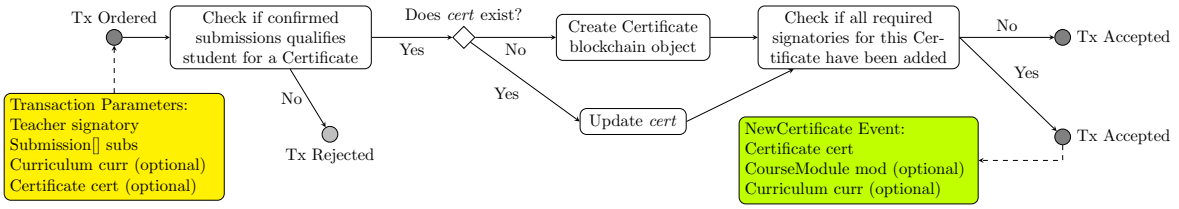

Fig. 5.10 Flowchart representation of the Smart Contract for the GenCertificate Transaction (Tx)

5.4.5 The ProposeCurriculum Transaction

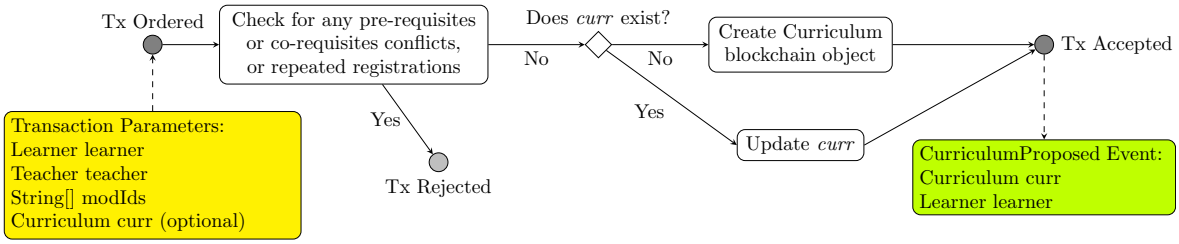

Fig. 5.11 Flowchart representation of the Smart Contract for the ProposeCurriculum Transaction (Tx)

5.4.6 The ApproveCurriculum Transaction

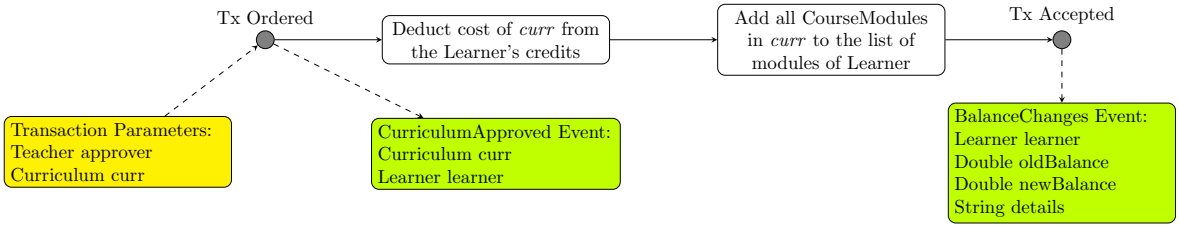

Fig. 5.12 Flowchart representation of the Smart Contract for the ApproveCurriculum Transaction (Tx)

5.5 Access Control

The design of access control on the blockchain network is based on the Access Control Language of Hyperledger Composer. It allows for the creation of access control rules that are conditional upon: the type of participant or asset, the use of a specific transaction, and the values of properties of the participant or asset.

This means the mode of access control for the blockchain is both role-based and attribute-based. Figure 5.13 is a generic representation of what an access control rule deployed to the Hyperledger Fabric blockchain is composed of, drawn based on the style proposed by Poniszewska-Maranda et al. (2005) for extended role-based access controls.

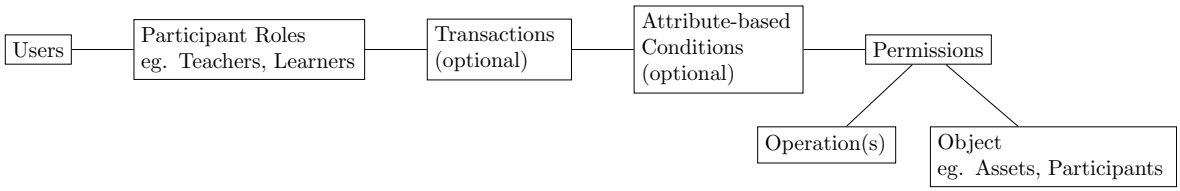

Fig. 5.13 A flowchart representation of the Access Control model offered by Hyperledger Composer (The Linux Foundation, 2018).

Most of the rules are a form of Mandatory Access Controls across the network, enforced by the system and cannot be modified by users or client applications, where access to objects is restricted based on fixed security attributes (Yuan and Tong, 2005). Unless explicitly allowed by a defined access control rule, all operations are denied by default.

Some rules are more discretionary because the security-related attributes can be changed by users. For example, a *Learner* will be able to set its own *acLevels* and *priviledgedReaderIds* fields to control what records related to the *Learner* that *readers* can read. Integer values are stored in *acLevels* correlating to a permission permutation model inspired by UNIX file permissions.

Table 5.1 The Reader access permutation model for Learner assets

	0	1	2	3	4	5	6	7
Certificates		✓		✓		✓		✓
Submissions			✓	✓			✓	✓
Learner, Curriculums					✓	✓	✓	✓

So an *acLevels* value of `[1, 3]` would mean all *readers* can read your *Certificates*, while *readers* included in *priviledgedReaderIds* can read your *Certificates* and *Submissions*.

Table 5.2 lists all of the access control rules designed for the blockchain of this project.

Table 5.2 The access control rules designed for the blockchain

	Role	Transaction	Attribute-based Condition	Operation(s)	Object(s)
1	Learner, Teacher, Reader	–	–	READ	CourseModule, ModuleUnit, Assessment
2	Learner	–	user.uId == object.uId	UPDATE, READ	Learner
3	Learner	–	–	READ	Teacher, Reader
4	Learner	AddSubmission	user.uId == object.learner.uId	CREATE	Submission
5	Learner	–	user.uId == object.learner.uId	READ	Submission
6	Learner	ProposeCurriculum	user.uId == object.learner.uId	CREATE, UPDATE	Curriculum
7	Learner	–	user.uId == object.learner.uId	READ	Curriculum
8	Teacher	–	user.uId == object.uId	UPDATE, READ	Teacher
9	Teacher	–	object.mods has (mod.teachers has (teacher.uId == user.uId))	READ	Learner
10	Teacher	–	–	READ	Reader
11	Teacher	CreateModule	object.teachers has (teacher.uId == user.uId)	CREATE	CourseModule, ModuleUnit, Assessment
12	Teacher	–	object.unit.mod.teachers has (teacher.uId == user.uId)	READ	Submission
13	Teacher	SubmitResult	user.uId == object.teacherAssigned.uId	UPDATE	Submission
14	Teacher	–	user.uId == object.teacher.uId	READ	Curriculum
15	Teacher	ProposeCurriculum, ApproveCurriculum	user.uId == object.teacher.uId	UPDATE	Curriculum

(Continued on next page)

	Role	Transaction	Attribute-based Condition	Operation(s)	Object(s)
16	Reader	–	user.uId == object.uId	UPDATE, READ	Reader
17	Reader	–	–	READ	Teacher
18	Reader	–	n = object.learner.acLevels[0]	READ*	Certificate (n>=1), Submission (n>=3), Learner, Curriculum (n>=5)
19	Reader	–	object.learner.priviledgedReaderIds has (id == user.uId) AND n = object.learner.acLevels[1]	READ*	Certificate (n>=1), Submission (n>=3), Learner, Curriculum (n>=5)
20	ADMIN	–	–	CREATE, UPDATE, DELETE	PARTICIPANTS
20	ADMIN	–	–	READ	ALL

5.6 Limitations

Some limitations have been discovered for the above blockchain network design. They have not been accounted for or patched because of either a desire to keep the design minimal and achievable, or a shortage of time to conduct background research. Here is a list of the notable limitations:

- The *Teacher* participant was not designed to have different tiers of privileges. In higher education, different roles such as module leader, module reviewer, supporting tutors exist. A module leader may also need the permissions to create new *Teacher* participants who are supporting tutors.
- There is no participant type for a higher education staff who is an institution administrator and not a teaching staff. They may require more permissions than a regular *Teacher* or *Reader*, and bespoke transactions.

5.7 User Interfaces for Client Applications

To demonstrate the improvements a blockchain back-end and Smart Contracts can bring to an e-Learning platform, a Learner client application and a Teacher client application will be built at the implementation stage.

While improving general usability is not one of the objectives of this project, poor usability and interface designs could adversely affect evaluation of the improvements that this project aims for.

Therefore, several of Nielsen (1995)'s famous five usability design goals and ten usability design hieuristics were considered when developing the application prototypes. See Table 5.3:

Table 5.3 The usability considerations for client applications

Usability Goal	Usability Hieuristic	Example Non-functional requirements
Learnability	Match between system and the real world	Using common higher education glossary
	Visibility of system status	Notifications Area
	Consistency and standards	Adopt a popular design language
Efficiency	Flexibility and efficiency of use	
Memorability	Recognition rather than recall	Always Visible Navigation Menu
Error Reduction	Help users recognize, diagnose, and recover from errors	Error Messages on Transaction Failures
Satisfaction	Aesthetic and minimalist design	

To maximise learnability and user familiarity, the client applications will adopt a popular design language for their user interfaces. Material Design, a design system that combines theory, resources,

and tools, is used by the most popular mobile operating system Android and many web applications (Google, 2018). Resources such as free-to-use icon packs and tools such as well-maintained user interface component libraries on various mobile and web development platforms will enable quick scaffolding of design compliant and highly usable applications.

Figure 5.14 are the sitemap trees showing the contextual flow for users in the client applications:

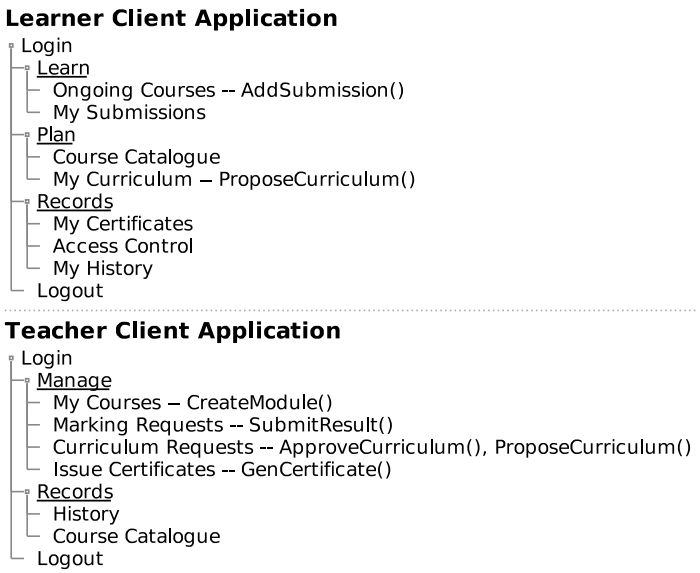

Fig. 5.14 Sitemap designs for the learner and teacher client applications with notes on the location of transaction ordering dialogues.

No low fidelity prototypes for the two demonstrator applications were produced due to the lack of research need, since the usability design was driven by hieuristics and not a more user-centred approach. The next chapter will describe further how high fidelity, demonstration-ready web applications were built for this project.

Chapter 6

Implementation

6.1 Development Environment and Tools

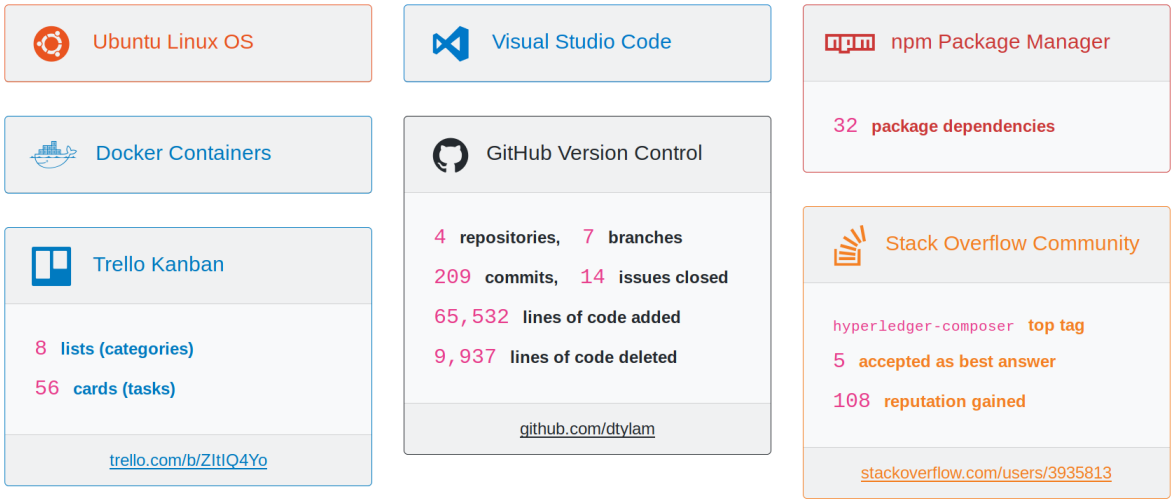

Fig. 6.1 A visual overview of the tools used and their usage statistics if any

Specific systems and tools were used to build the demonstrator network and client applications (as in Figure 6.1). These choices and their rationale are detailed below:

Operating System: The developer's guide for both Hyperledger Composer and Hyperledger Fabric recommends using Ubuntu or Mac OS as the host operating system for development. Ubuntu is selected for being the free and open source option. The development personal computer, which was originally a Windows machine, was set up to dual boot with the latest Ubuntu release.

Version Control: A version control system or software keeps track of source code modifications, so that developers can compare earlier versions of the code, revert changes, and minimise disruptions of mistakes (Atlassian, 2018). It is essential to medium to large scaled projects.

All work done at the implementation stage was tracked with the version control system Git. Git is a distributed version control system, where repositories can be backed up to a remote server, such as on the cloud. This is done with GitHub, a git-based version control, code hosting and project management service that offers free private repositories to verified students (GitHub, 2018).

Code Editor: The Hyperledger Composer framework does not require a dedicated integrated development environment and recommends using a text editor. Visual Studio Code, an open source text editor developed by Microsoft was chosen as it has a dedicated official syntax checking and beautifying plugin for Hyperledger Composer.

Community Support: Hyperledger Composer and Hyperledger Fabric have active communities on Stack Overflow, a popular online community forums for developers to discuss coding problems. Throughout the duration of the project, it has been a source of solutions to common problems faced and solved by many other users. Towards the end of the project, answer contributions were also made.

Several other tools were also used, for example, as previously mentioned in Chapter 3, Trello was used to manage feature prioritisation for the agile development process. The use of Docker containers and npm package manager will be explained in due course below.

6.2 Architecture and Tech Stack

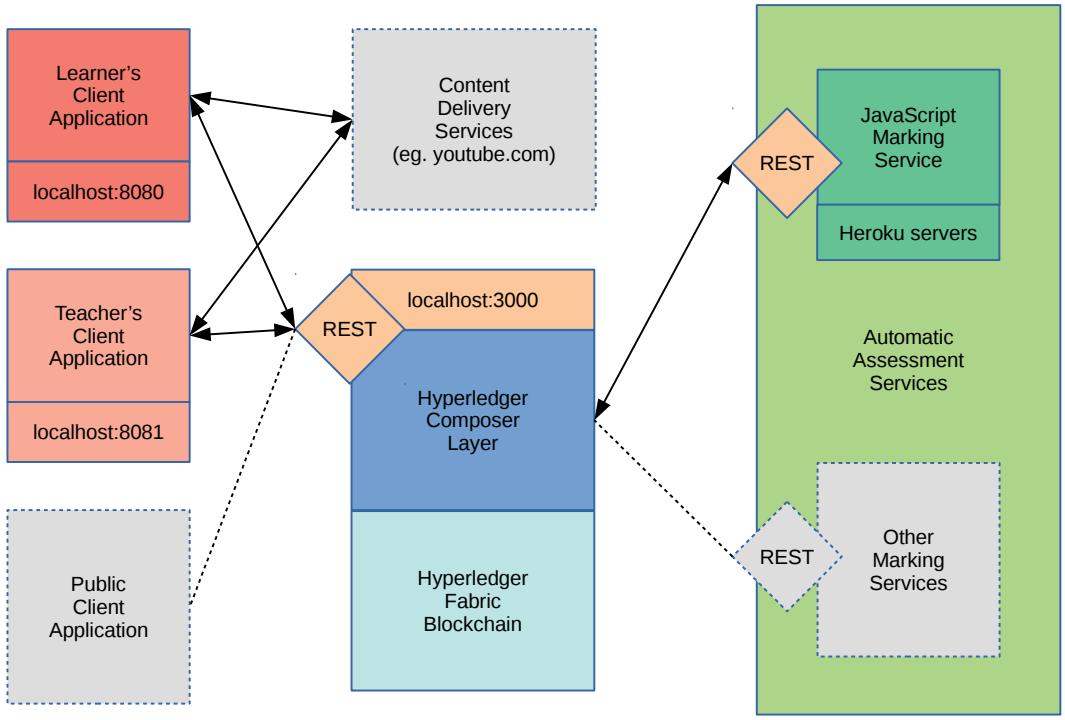

Fig. 6.2 Component architecture overview for the demonstrator system built (dashed lines components were not built)

The microservices architecture pattern is used to design the demonstrator system. Microservices is a pattern proposed by Richardson (2018) that structures a system as a set of loosely coupled, collaborating services, partitioned so that each service corresponds to a business capability.

In the demonstrator system (as in Figure 6.2), this is exemplified by the separation of teacher and student applications, the separation of content delivery and student records (blockchain), and the separation of different automatic marking services. They communicate through RESTful web protocols (HTTP and JSON). This also enables the use of different technologies, separate testing and deployment for each component.

Figure 6.3 shows an alternate overview of the demonstrator system with its technology stack. These main software packages are described further below.

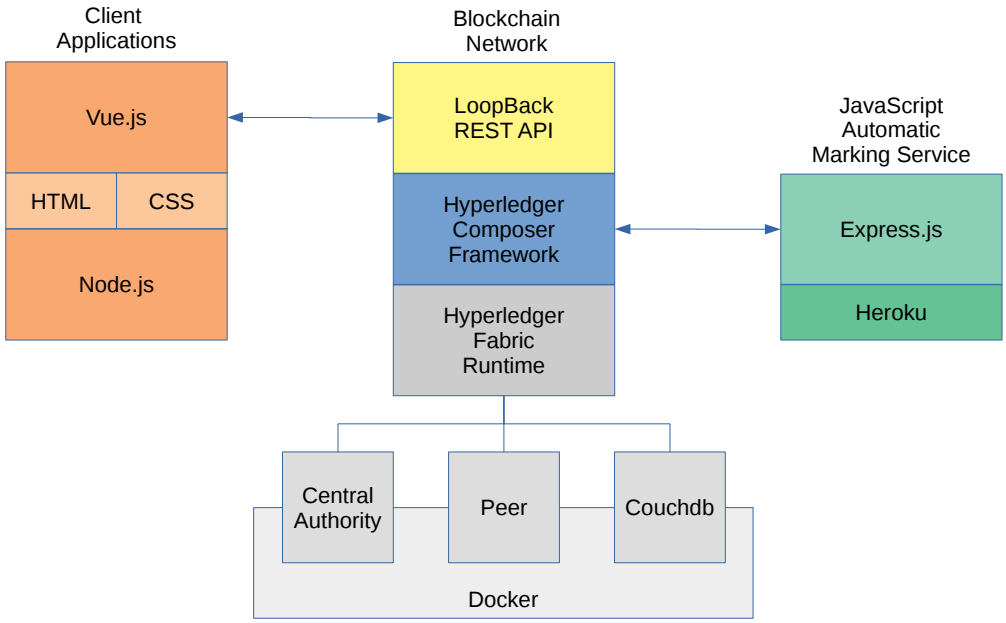

Fig. 6.3 Technology Stack overview for the demonstrator system built

6.2.1 Client Applications

The client applications for learners and teachers has to be highly dynamic (non-static) web applications, with the JavaScript scripting language facilitating most of the user interactions.

Vue.js is a JavaScript-based framework for developing web interfaces. Vue.js was chosen over other popular frameworks such as React.js and Angular.js because it is focused on the view layer and is much simpler and faster to learn. The Node.js JavaScript runtime is used to run the Vue.js applications. The npm package manager is used to install and manage the packages used. See more details such as version numbers in Appendix B.1.

6.2.2 Blockchain Network

Hyperledger Composer deploys to Hyperledger Fabric, which runs on Docker. Docker is a containerisation platform used to develop distributed systems. A container is a lightweight, stand-alone package of a piece of software that includes everything needed to run it from code and tools to system settings.

Docker can emulate multiple servers and run them separately on one development machine. This provides the peers network needed to host the distributed ledger required for the Hyperledger Fabric blockchain. A minimum of three docker containers are required to start a Fabric blockchain: the Central Authority server, a minimum of one peer server, and a Couchdb server, which is a database that stores the state of the blockchain in a queryable form.

See the full list of pre-requisites and dependencies in in Appendix B.2.

6.2.3 Automatic Marking Service

More automatic marking services were originally planned but due to time constraints only a simple JavaScript server that checks for file equivalence was implemented.

This was written with Express.js, a minimal Node.js web application framework and hosted on Heroku, a Node.js capable cloud platform with a free tier. See the full list of pre-requisites and dependencies in in Appendix B.3.

6.3 Blockchain Network Development

Beginning here, progress descriptions and evidences of implementation work will be presented.

6.3.1 Command Line Interface

Using Docker, setup images of Hyperledger Fabric were downloaded, the basic three servers were set up and a certificate for the central authority administrator was generated. This allows Hyperledger Composer to later authenticate as the blockchain administrator and import custom network definitions.

The Hyperledger Composer notations from the design phase where packaged into a business network archive (.bna) file, and deployed to the Hyperledger Fabric blockchain.

Three bash shell scripts were written:

1. *start.sh*: this script starts Hyperledger Fabric, generate the minimum number of peers, and connects as the blockchain administrator. It then creates the .bna file with Hyperledger Composer and deploys it to the Fabric blockchain. See Figure 6.4 for a screenshot of the script running.

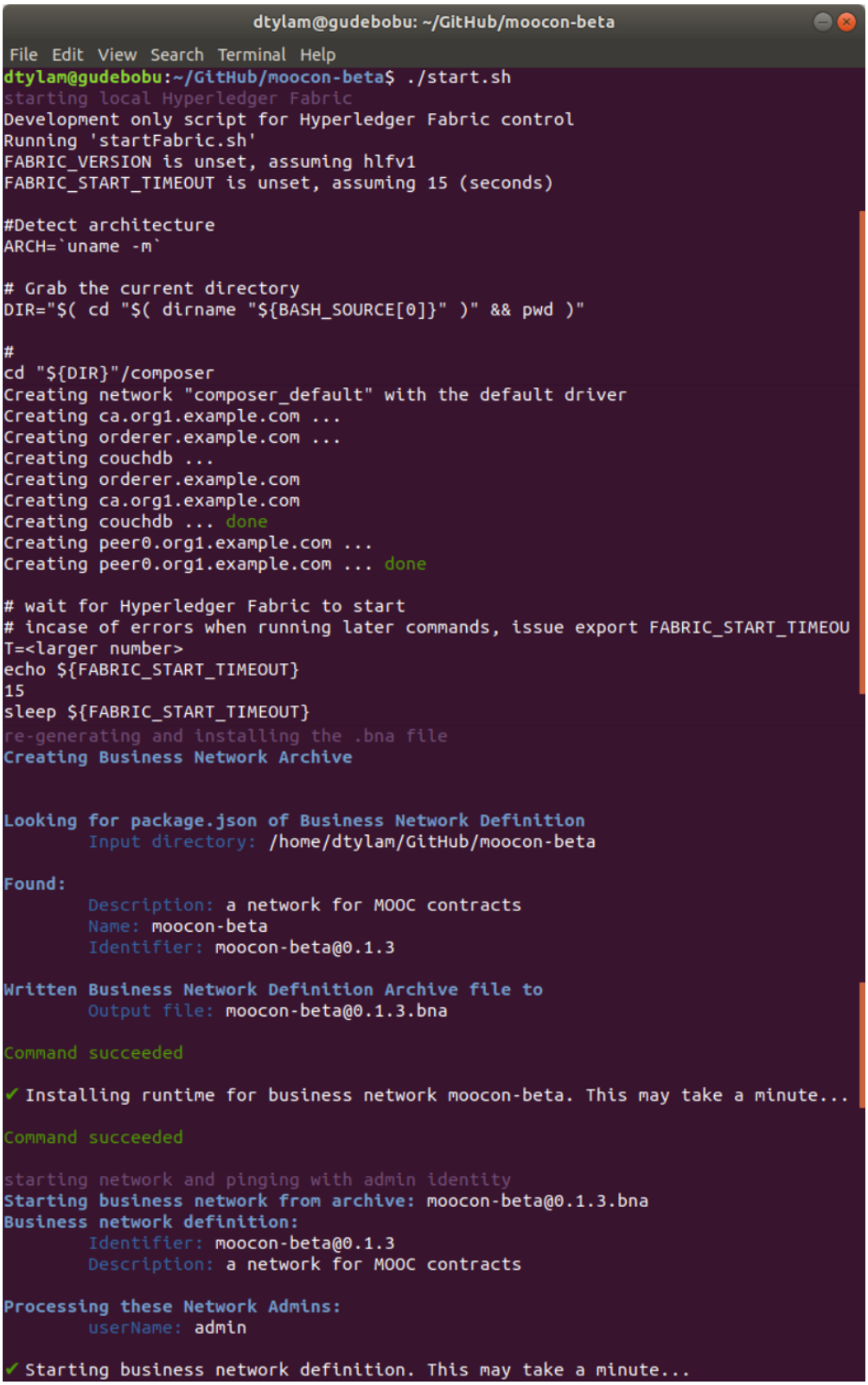

Fig. 6.4 Screenshot of the start.sh script in action in a command line interface

2. *load_participant.sh*: this script connects as the blockchain administrator and creates three *Learner*, four *Teacher* and two *Reader* participants on the blockchain. A .card file is created for each participant, which contains the certificate they need to connect to the blockchain as

themselves. In the final stages of development, this script is called by the previous *start.sh* to simplify the starting and loading process. See Figure 6.5 for a screenshot of the script running.

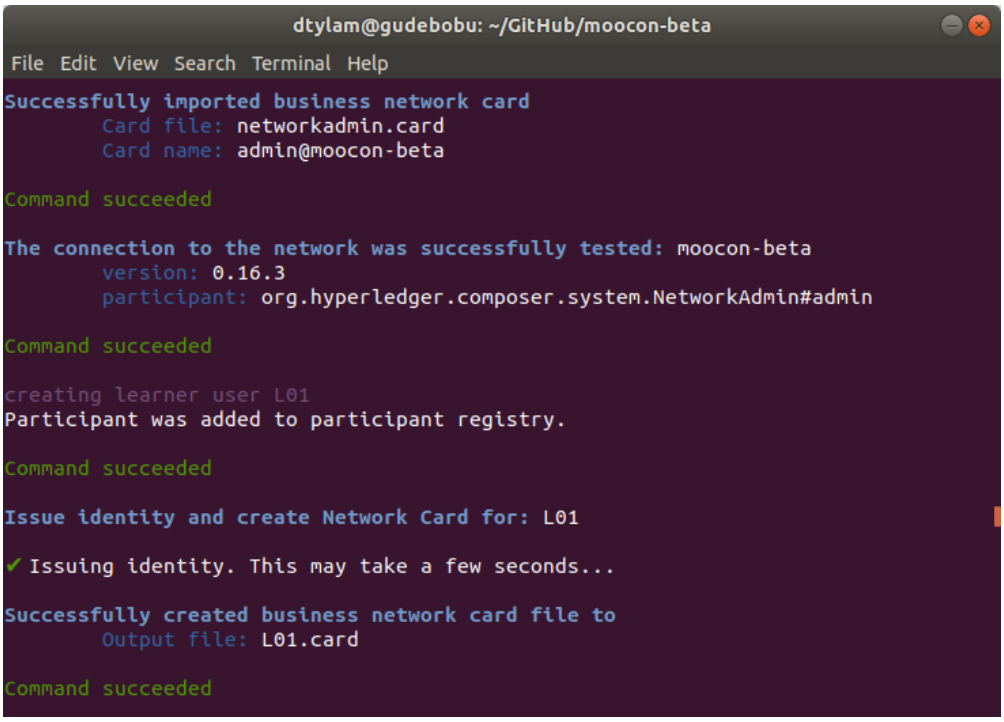

Fig. 6.5 Screenshot of the load_participant.sh script in action in a command line interface

3. *destroy.sh*: at the end of every development session the blockchain is completely erased and removed. This is because unwanted data entries or transactions may have occurred, and the blockchain schema can only be changed with a hard fork/ restart. See Figure 6.6 for a screenshot of the script running.

The GitHub source code repository used for the blockchain schema and command line scripting is hosted at github.com/dtylam/moocon-beta. Three branches (significant iterations) were created for this component, below in chronological order:

1. *moocon-alpha* branch: this is an archived repository of design work done with version 0.15.x of Hyperledger Composer, the project upgraded to version 0.16 in December, which contained breaking changes. The 0.16 upgrade was recommended by the Hyperledger developers because it is a long term support release that is feature complete and stable, and adds the useful .card file authentication protocol.
2. *master* branch: this is the original working branch on Hyperledger Composer 0.16, a bulk of design and development work was done, and changes to the schema and transactions were made based on literature review and background research.

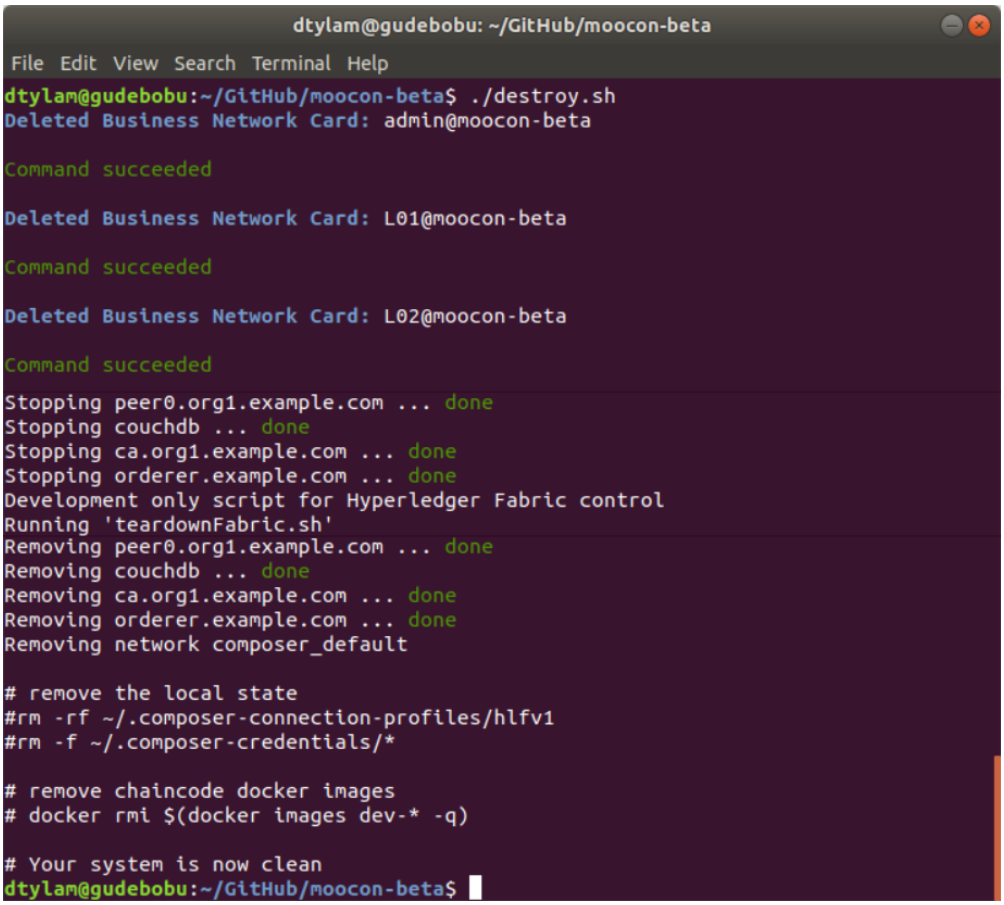

Fig. 6.6 Screenshot of the destroy.sh script in action in a command line interface

3. *bcu* branch: this is the final, demonstration ready branch named after the proposed marketplace "Blockchain University". Changes were made to satisfy new requirements obtained through primary data (interviews in Chapter 4).

6.3.2 Application Program Interface

A readily available script connects the deployed blockchain to Loopback, an open source Node.js RESTful Application Program Interface (API) framework. The command used is:

```
composer-rest-server --card admin@moocon-beta --namespaces always
--websockets true --port 3000 --authentication true --multiuser true
```

This hosts a production-ready API at `localhost:3000`, connected to the server of the central authority administrator. Figure 6.7 shows the API explorer view generated by Loopback at `localhost:3000/explorer`.

Limitations exist for how multiple participants can use this same API. The API is authenticated as one peer at one time. It uses a construct called `wallet` to store .card files and switches between them

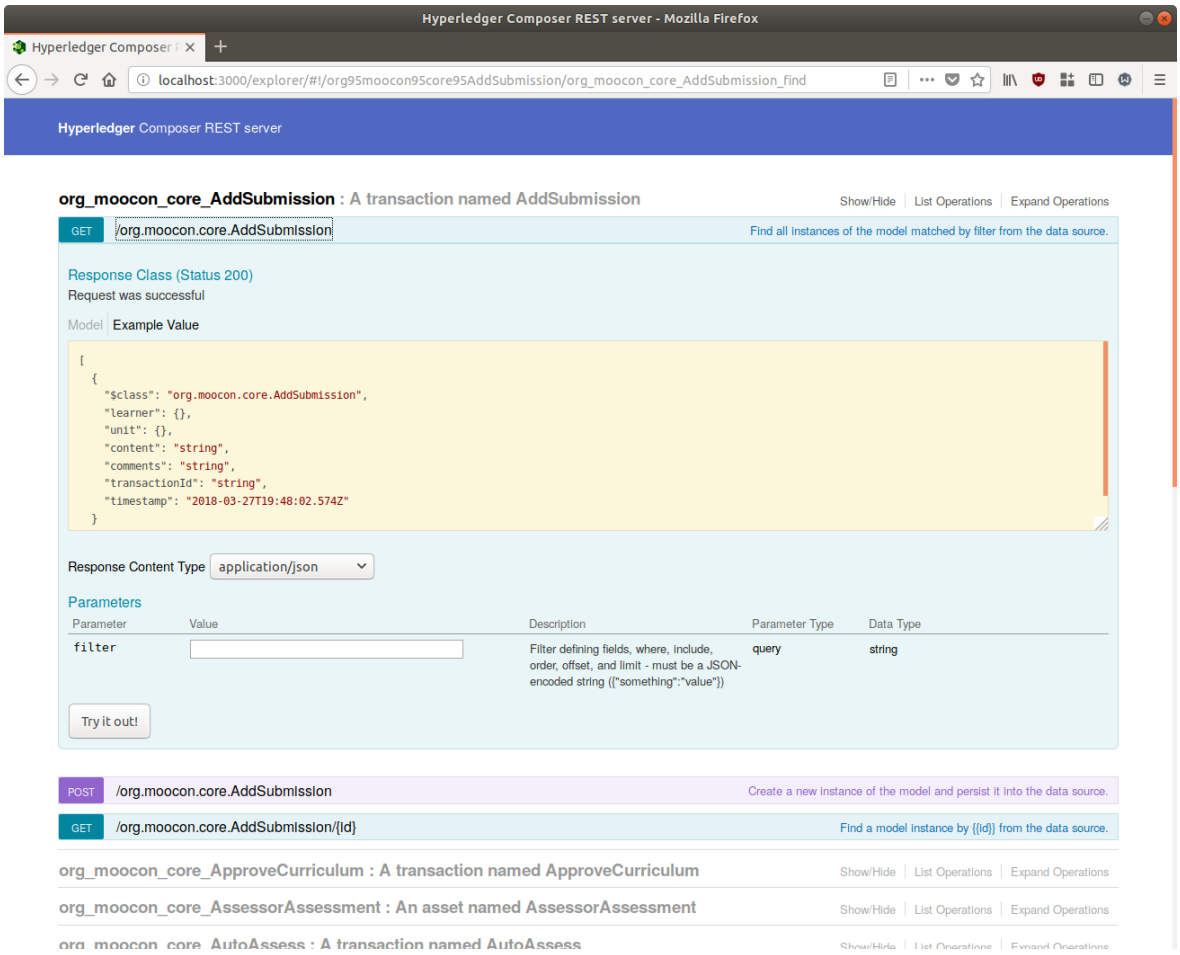

Fig. 6.7 Screenshot of the auto-generated Loopback Explorer for the blockchain according to Hyperledger Composer definitions

to create a "multiuser" API. So to switch between users in the client applications, a request must be made to `localhost:3000/wallet/{cardFileName}/setDefault`. This limitation is accepted for the demonstrator system, as it does not impact core requirements.

6.4 Automatic Marking Service Development

The first iteration of the Blockchain Network has no automatic marking Smart Contract features. This is due to a limitation found during development of the original design. The original design of the Automatic Assessment Smart Contract would have the Smart Contract run a custom block of chaincode created by a Teacher for each assessment (See left hand side of Figure 6.8).

However, the Hyperledger Composer framework does not allow chaincode to run code foreign to its main packages and vanilla JavaScript. This is due to the transpilation and deployment of Hyperledger Composer chaincode to the Hyperledger Fabric blockchain, a process which ordinary developers have

no convenient controls over. It will not be able to accept new snippets of code, created and uploaded to the blockchain by Teacher participants.

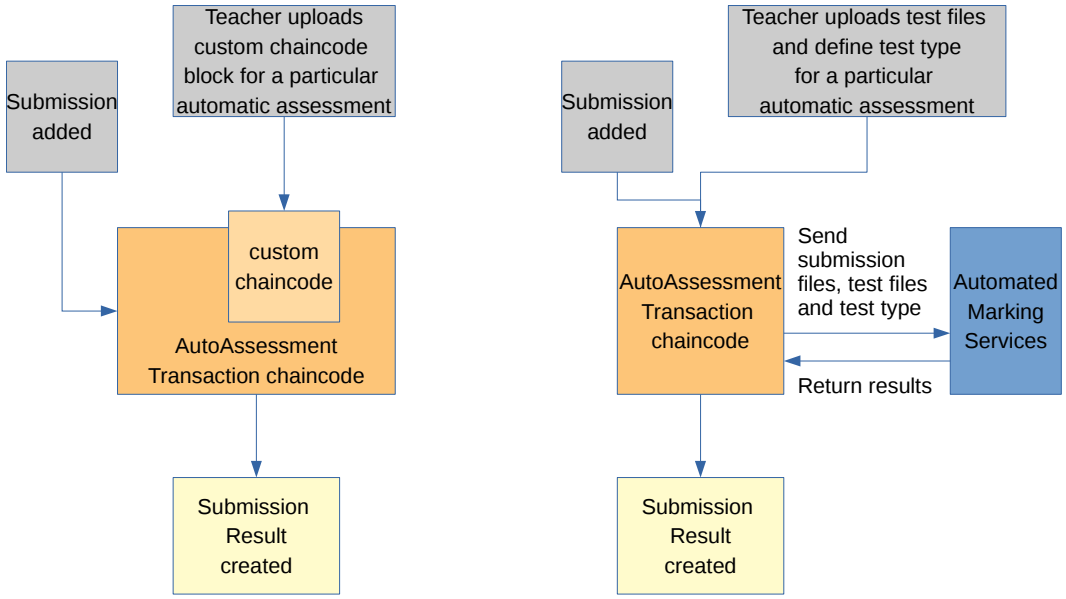

Fig. 6.8 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

Understanding this limitation and after studying documentations further, a technically informed design was created (See right hand side of Figure 6.8). he Hyperledger Composer framework allows chaincode to communicate with external APIs through a built-in method `post(url,data)` where `url` is the endpoint address of the desired API and `data` is a blockchain asset or concept serialised into JSON format.

This limitation and subsequent redesign has actually inspired the microservices architectural approach covered in the previous section 6.1.

To use the `post(url,data)` functionality effectively, an extra asset *AutoAssessRequest* was added to the blockchain schema. *AutoAssessRequest* would contain the submission content file, the test file on the blockchain for this assessment, the test type of this assessment (eg. software testing, AI testing, etc), and an API response field. This packages all the information the external Automated Marking Services will need, and keep the request and response records permanently on the blockchain.

To build an example automated marking service, a string equivalence test was created on a simple Express.js application server. Here is the simple code snippet used:

```
if (req.body.testFile === req.body.content) {
    res.json(true); // if the test file is equal to the submission file
} else { res.json(false); }
```

Even though this is just a simple equivalence test, it proves the blockchain's ability to conduct automated assessments, and more sophisticated assessment APIs could be called or routed to.

It is essential that this testing service is available on the world-wide web, because the peers are contained in Docker containers where the normal workstation `localhost` address is not easily reachable. Therefore, the heroku Node.js hosting service is used and this example automated marking service is available at moocon-js-marking.herokuapp.com/.

The source code of this component is also tracked and hosted at github.com/dtylam/moocon-js-marking.

6.5 Client Applications Development

Before developing the client applications, some example data was created and imported into the network. This was nine courses modules, their module units and assessments. These data were created by loosely following the syllabuses of courses from Massive Open Online Courses.

A video demonstration showcasing the user journeys across both applications was created and available on youtube.com/FUMBn6wPG5M.

An open source styling and component library, Vue-material, was used to create Material Design compliant user interfaces. Notes on the development process are further detailed below.

6.5.1 Learner Application

The source code of this application is tracked and hosted at github.com/dtylam/moocon-learner-client.

The switch from traditional static webpage development to component-driven development was a steep learning curve. Traditional web development focuses on the separation of concerns by file types, where markup (HTML), styling (CSS) and scripting (JavaScript) are in different files and folders.

JavaScript-based web applications, on the other hand, encourage organisation according to functional components. In Vue.js for example, components (.vue) files are used to write HTML, CSS and JavaScript in one file (Vue.js, 2017), which creates a component in the web application. It is afterwards packaged to be production-ready with build tools.

For example, the entire web application only uses one main `index.html` page. Each main menu button such as "Ongoing Courses" swaps out the previous page component for another. Components can also be nested, the YouTube embedding view is its own component, for example.

Below are some annotated screenshots of the application (Figures to).

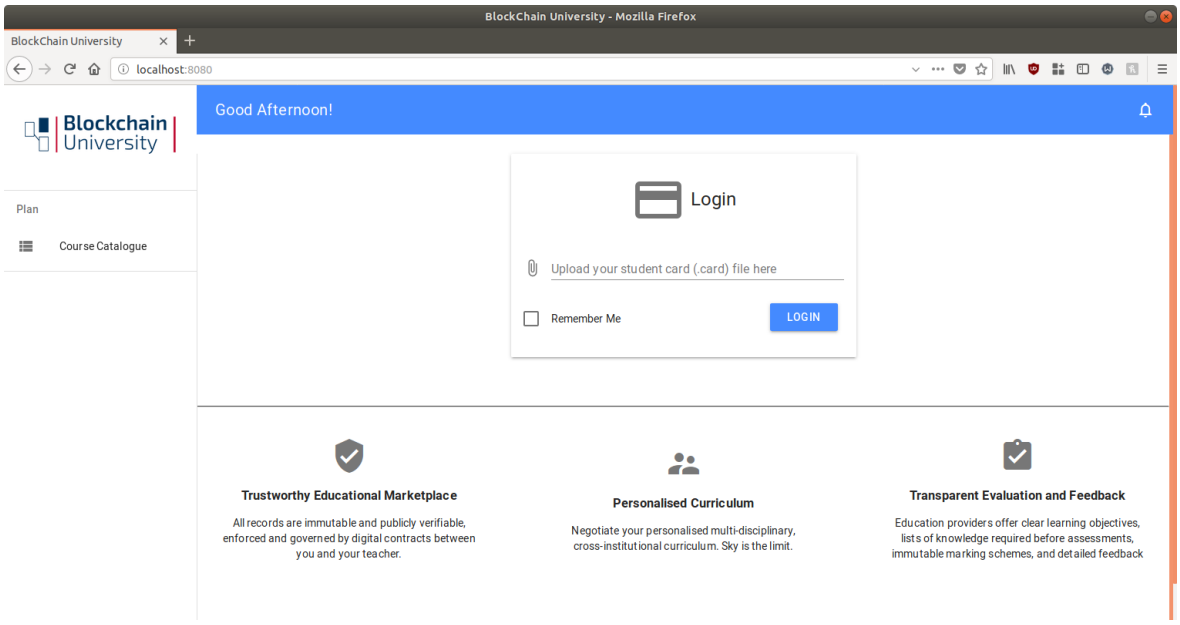

Fig. 6.9 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

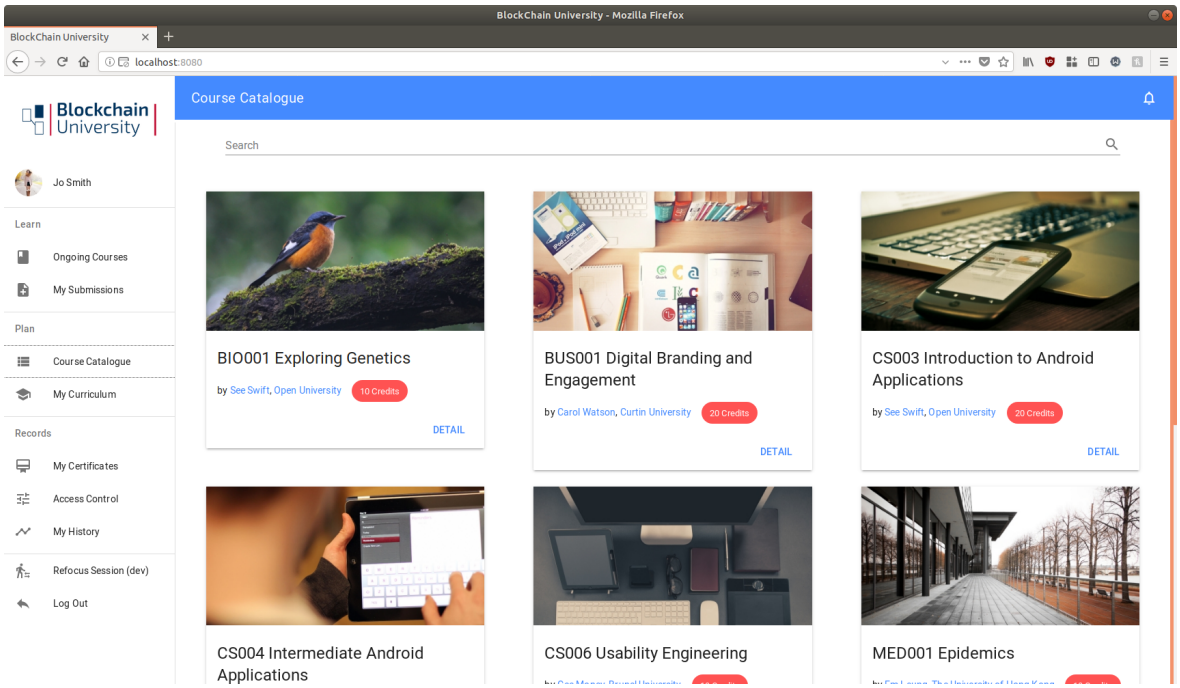

Fig. 6.10 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

6.5.2 Teacher Application

The Teacher application was created as a fork of the Learner application. The source code of this application is tracked and hosted at github.com/dtylam/moocon-teacher-client.

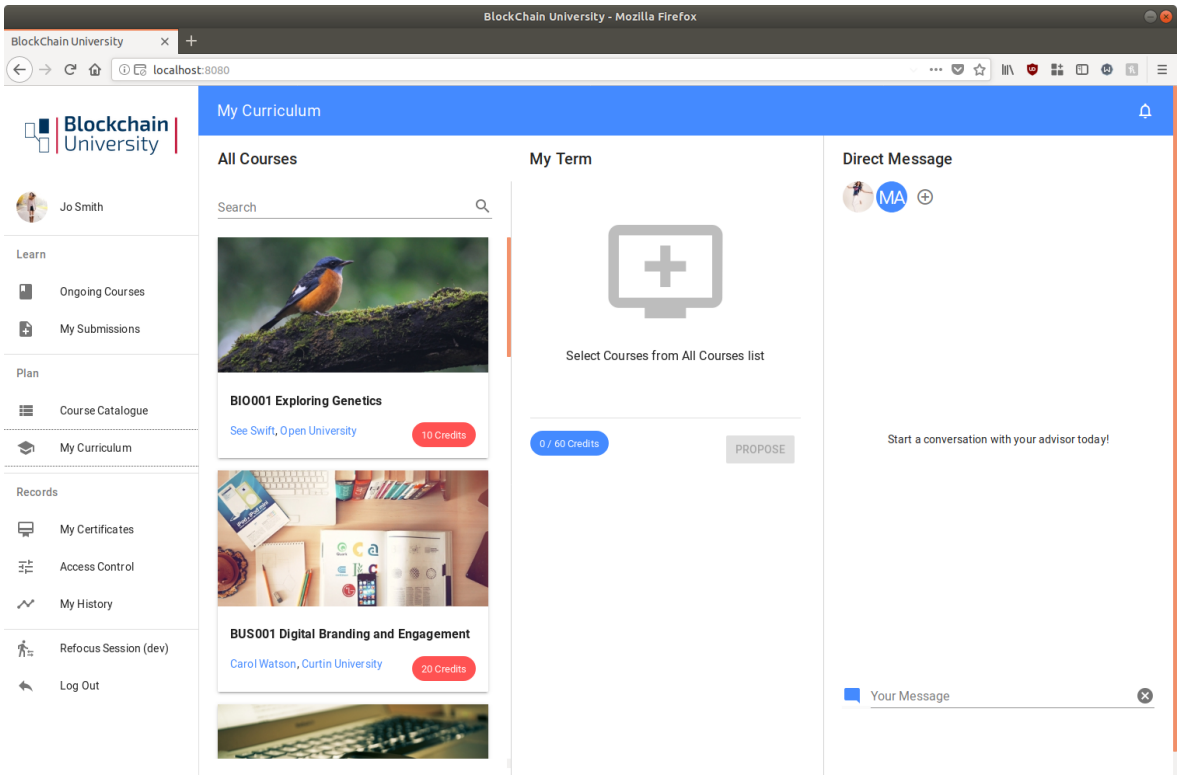

Fig. 6.11 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

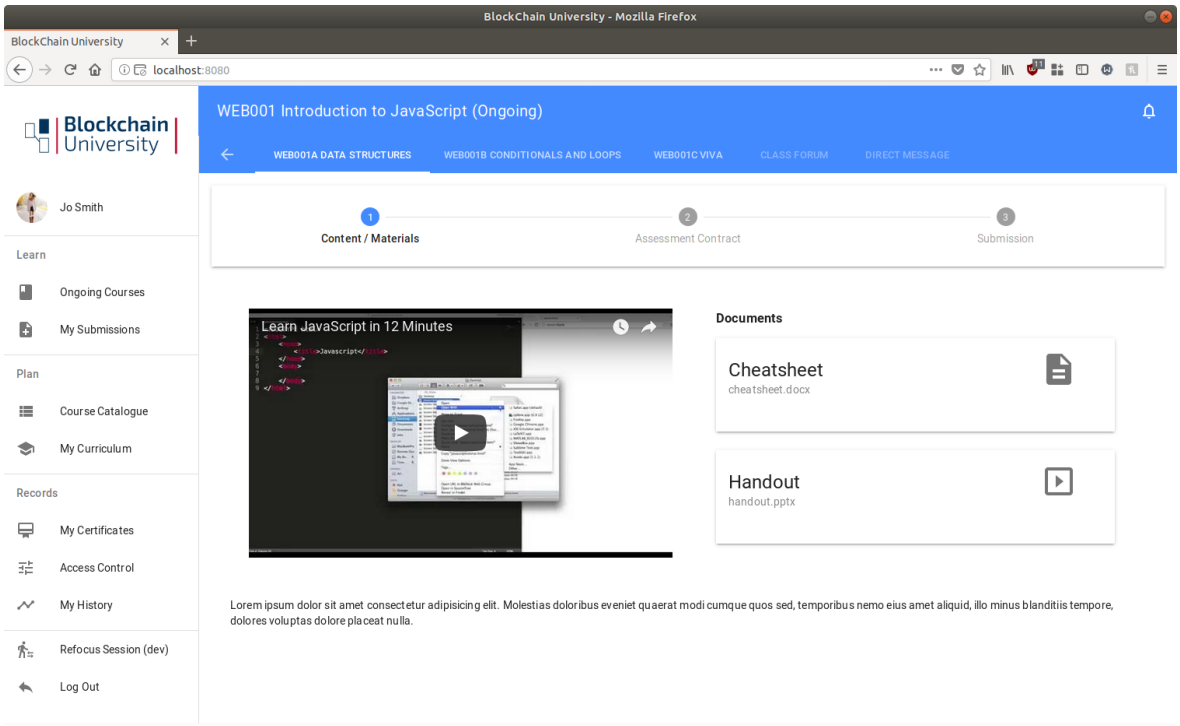

Fig. 6.12 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

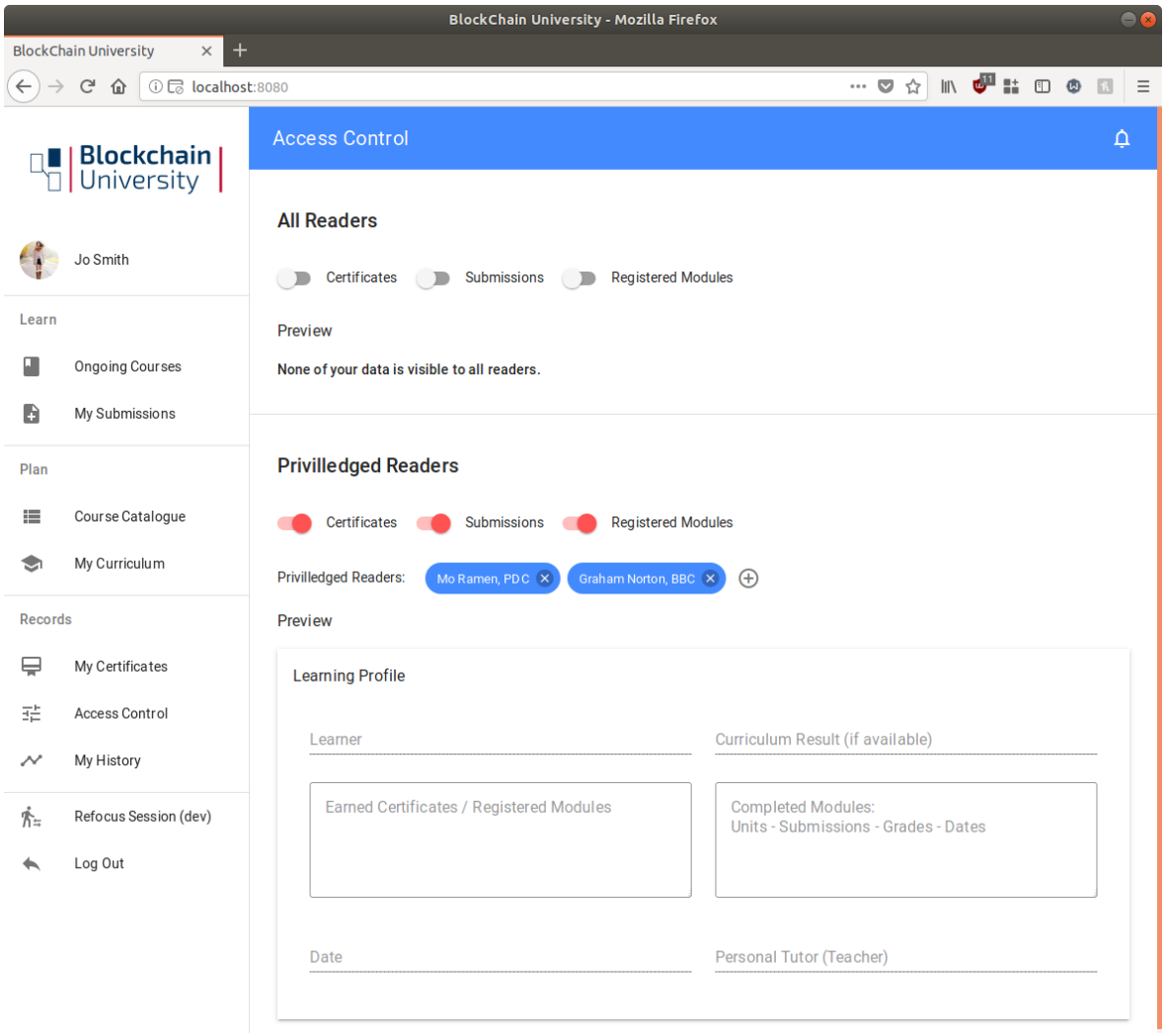

Fig. 6.13 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

6.6 Test Driven Development

mocha

postman

6.7 Limitations and Issues

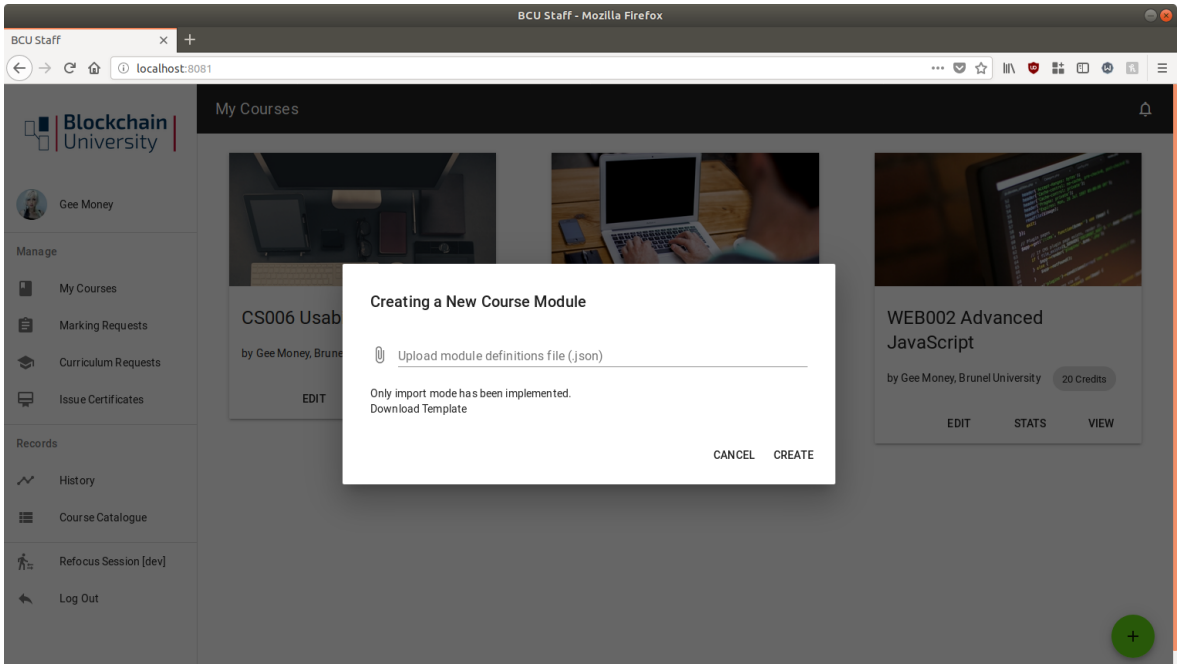

Fig. 6.14 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

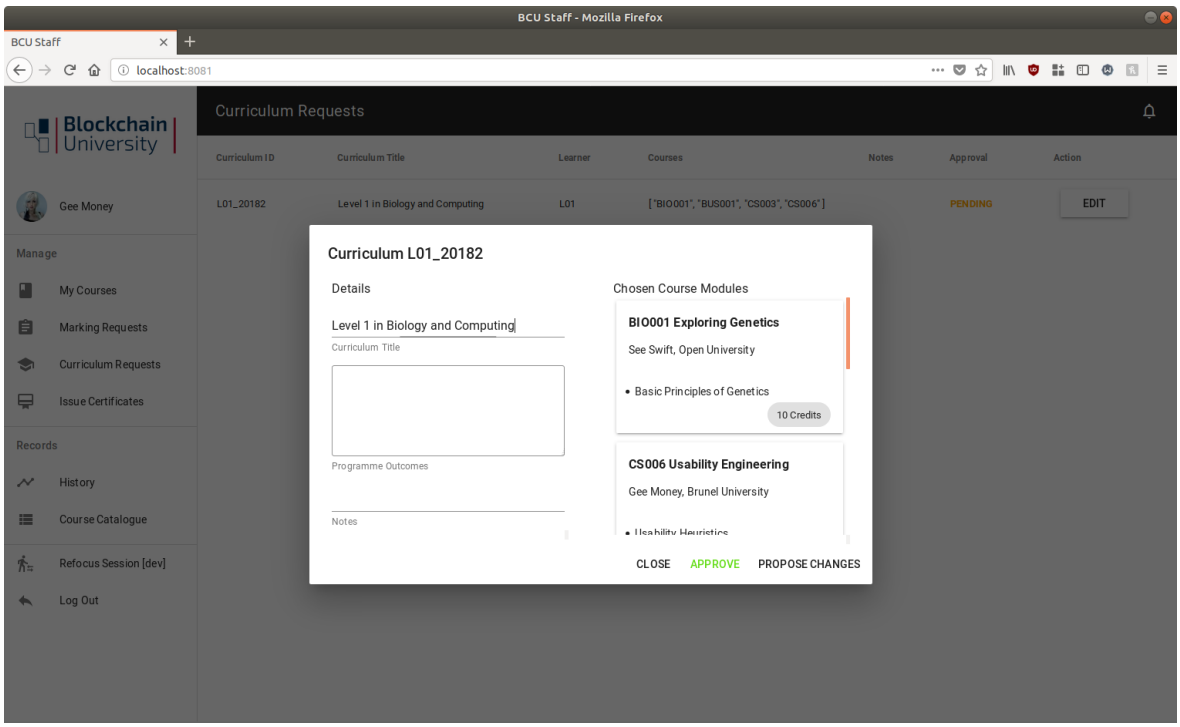

Fig. 6.15 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

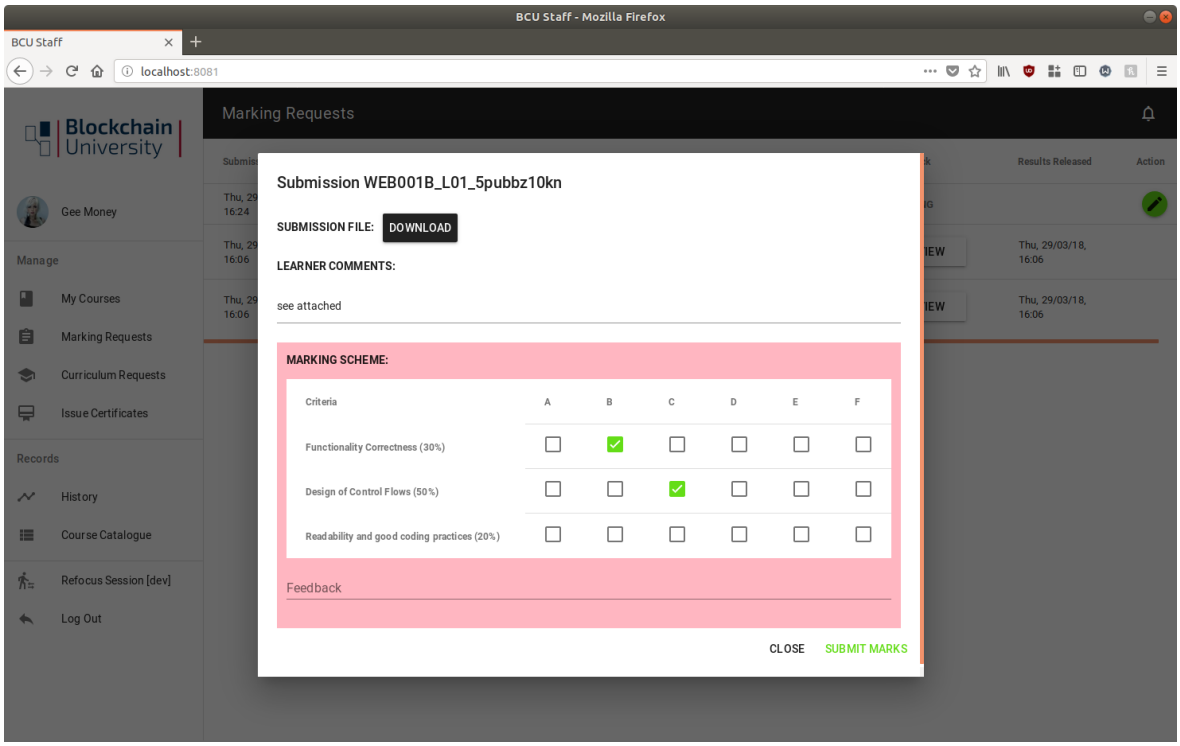

Fig. 6.16 Flowchart of original design (left) and the technically informed design (right) of how the Automatic Assessment Smart Contract should run

Chapter 7

Evaluation

purpose of eval

7.1 Methodology

instruments: appdx

sample

7.2 Testing

7.3 Interviews with Education Professionals

7.4 Interviews with Student Representatives

7.5 Analysis

7.6 Conclusion

Chapter 8

Conclusion

8.1 Future Work

- tests embedded in smart contracts instead of rest calls, which may not always be available

- consensus model for double marking, etc

- the higher education community explicit description of learning outcomes, programme specifications, grade descriptors is not enough engaging students in activities pre-assessment intervention (Bryan and Clegg, 2006)

and here I write more . . .

https://www.timeshighereducation.com/news/oxford-academics-launch-worlds-first-blockchain-university

References

Agile Business Consortium (2018). Moscow prioritisation | the dsdm agile project framework (2014 onwards). [online] https://www.agilebusiness.org/content/moscow-prioritisation.

Ali, S. (2005). Effective teaching pedagogies for undergraduate computer science. *Mathematics and Computer Education*, 39(3):243.

Atlassian (2018). What is version control. [online] https://www.atlassian.com/git/tutorials/what-is-version-control.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al. (2001). Manifesto for agile software development. [online] http://agilemanifesto.org/principles.html.

Bessonov, A. (2017). Five trends impacted by blockchain technology. [online] https://www.forbes.com/sites/forbestechcouncil/2017/09/20/five-trends-impacted-by-blockchain-technology.

blockcerts.org (2018). Faq - blockcerts : The open initiative for blockchain certificates. [online] https://www.blockcerts.org/guide/faq.html.

Bragg, S. (2014). Education, 'consumerism' and 'personalisation'. 35.

Brown Jr, J. (1999). *Assessment matters in higher education*. McGraw-Hill Education (UK).

Bryan, C. and Clegg, K. (2006). *Innovative assessment in higher education*. Routledge.

Campbell, A. (2010). Digital forms of assessment: Assessing what counts, the performance. *Curriculum, technology & transformation for an unknown future. Proceedings ascilite Sydney*, 2010.

Condie, R. and Munro, B. (2007). The impact of ict in schools: Landscape review.

Cuicapuza, J. (2017). Hyperledger's fabric composer: Simplifying business networks on blockchain. [online] https://medium.com/@RichardCuica/hyperledgers-fabric-composer-simplifying-business-networks-on-blockchain-94313b979671.

Denham, E. (2017). Gdpr – sorting the fact from the fiction. [online] https://iconewsblog.org.uk/2017/08/09/gdpr-sorting-the-fact-from-the-fiction/.

Dron, J. (2007). Designing the undesignable: Social software and control. *Educational Technology & Society*, 10(3):60–71.

El-Khatib, K., Korba, L., Xu, Y., and Yee, G. (2003). Privacy and security in e-learning. *International Journal of Distance Education Technologies (IJDET)*, 1(4):1–19.

GitHub, I. (2018). Github education. [online] https://education.github.com/.

Google (2018). Material design. [online] https://material.io/.

Green, H., Facer, K., Rudd, T., Dillon, P., and Humphreys, P. (2005). Futurelab: Personalisation and digital technologies.

Gruber, J. (2004). Daring fireball: Markdown. [online] https://daringfireball.net/projects/markdown/.

Gulhane, I. (2017). Create and execute blockchain smart contracts - ibm code.

IBM Blockchain (2018). Ibm blockchain based on hyperledger fabric from the linux foundation. [online] https://www.ibm.com/blockchain/hyperledger.html.

IEEE Computer Society (2013). Ieee standard for learning technology—learning technology systems architecture (ltsa). *IEEE Std*, 1484.1.

Middleton, P. and Joyce, D. (2012). Lean software management: Bbc worldwide case study. *IEEE Transactions on Engineering Management*, 59(1):20–32.

Nielsen, J. (1995). 10 usability heuristics for user interface design. *Nielsen Norman Group*, 1(1).

Open University (2018). Open blockchain. [online] http://blockchain.open.ac.uk/.

Opsecmonkey (2017). Edmodo education platform hit by hacker, 77m account details stolen. [online] http://opsecmonkey.com/edmodo-education-platform-hit-by-hacker-77m-account-details-stolen/.

Pantò, E. and Comas-Quinn, A. (2013). The challenge of open education. *Journal of E-learning and Knowledge Society*, 9(1):11–22.

Paulsen, M. F. (2004). Online education and learning management systems–global e-learning in a scandinavian perspective. *Education Review//Reseñas Educativas*.

Poniszewska-Maranda, A., Goncalves, G., and Hemery, F. (2005). Representation of extended rbac model using uml language. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 413–417. Springer.

Richardson, C. (2018). Microservice architecture pattern. [online] http://microservices.io/patterns/microservices.html.

Sharples, M. and Domingue, J. (2016). The blockchain and kudos: A distributed system for educational record, reputation and reward. In *European Conference on Technology Enhanced Learning*, pages 490–496. Springer.

Sony Global Education (2017). Sge education blockchain. [online] https://blockchain.sonyged.com/.

Suhre, C. J., Jansen, E. P., and Torenbeek, M. (2013). Determinants of timely completion: the impact of bachelor's degree programme characteristics and student motivation on study progress. *Higher Education Research & Development*, 32(3):479–492.

Swan, M. (2015). *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.".

The Linux Foundation (2018). Introduction | hyperledger composer. [online] https://hyperledger.github.io/composer/introduction/introduction.html.

Valenta, M. and Sandner, P. (2017). Comparison of ethereum, hyperledger fabric and corda. Technical report, FSBC Working Paper.

Vue.js (2017). Single file components. [online] https://vuejs.org/v2/guide/single-file-components.html.

Walport, M. (2016). Distributed ledger technology: beyond block chain. *UK Government Office for Science*.

Wüst, K. and Gervais, A. (2017). Do you need a blockchain? *IACR Cryptology ePrint Archive*, 2017:375.

Xu, X., Pautasso, C., Zhu, L., Gramoli, V., Ponomarev, A., Tran, A. B., and Chen, S. (2016). The blockchain as a software connector. In *Software Architecture (WICSA), 2016 13th Working IEEE/IFIP Conference on*, pages 182–191. IEEE.

Yuan, E. and Tong, J. (2005). Attributed based access control (abac) for web services. In *Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on*. IEEE.

Appendix A

Reflection

Appendix B

Demonstrator System Dependencies

B.1 Client Applications

1. **vuejs**: vue (2.5.11), vue-loader (13.0.5), vue-template-compiler (2.4.4)
   the web development UI framework used
2. **vue-material** (1.0.0-beta-7)
   a styling framework for Vue.js that follows the Google Material Design specifications
3. **vue-config** (1.0.0)
   a plugin for Vue.js that allows simple caching of some configuration variables
4. **vue-youtube-embed** (2.1.3)
   a plugin for Vue.js that makes embedding videos from youtube.com easier
5. **babel**: babel-core (6.26.0), babel-loader (7.1.2), babel-preset-env (1.6.0), babel-preset-stage-3 (6.24.1)
   a tool that helps you write code in the latest version of JavaScript, compiling written code down to supported version if the environment does not run the latest.
6. **webpack**: webpack (3.6.0), webpack-dev-server (2.9.1), css-loader (0.28.7), file-loader (1.1.4)
   a automatic bundling tool for developing JavaScript applications, transforming and organising assets such as images and CSS styling files

B.2 Blockchain Network

Hyperledger Fabric Pre-requisites:

1. **Ubuntu Linux** (17.10)
2. **Docker Engine** (17.12)
3. **Docker-Compose** (1.13.0)
4. **Node** (8.9.1)
5. **npm** (5.6.0)
6. **Python** (2.7.14)

Hyperledger Composer Pre-requisites and Dependencies:

1. **Node** (8.9.1)
2. **npm** (5.6.0)
3. **hyperledger composer** (0.16.3)
4. **mocha** (3.2.0)
   javascript test framework for node.js
5. **chai** (3.5.0)
   test driven development assertion framework for node.js
6. **moment** (2.17.1)
   a package that help with parsing and displaying dates in JavaScript
7. **jsdoc** (3.5.5)
   an API documentation generator

B.3 Automatic Marking Service

JavaScript Marking Service Dependencies:

1. **expressjs**: express (4.15.5), body-parser (1.18.2), serve-favicon (2.4.5), morgan (1.9.0)
2. **heroku-cli** (6.15.39)
3. **Node** (9.9.0)
4. **npm** (5.6.0)