

Knowledge-related processes
critical to the enabling of
systematic software asset reuse
in a global IT company

Kathleen Wolfram

Submitted for the degree of
Doctor of Business Administration

Heriot-Watt University
Edinburgh Business School

February 2024

The copyright in this thesis is owned by the author. Any quotation from the thesis or use of any of the information contained in it must acknowledge this thesis as the source of the quotation or information.

Abstract

This research presents a case study on the knowledge management processes critical for achieving systematic software asset reuse in a global IT company. Reusing a software asset and its related artefacts at selected other business clients drives innovation, increases efficiency and can generate several million dollars in revenue from just one reuse. To date, known software asset reuse success is limited. Despite practical relevance, research has stagnated by mainly investigating technical reuse aspects.

This thesis addresses three gaps in the literature by looking from a business perspective at the intellectual capital required for software asset reuse, presenting five real-life software asset reuses and detailing the knowledge management processes towards systematic software asset reuse. The answers to the research questions advance the academic literature in the fields of intellectual capital, circular economy and knowledge.

Theoretical implications are: The intellectual capital required for systematic software asset reuse is a particular software asset, the reusable software asset, further defined here. The circular economy is enriched by adding a two-step distribution task to the reuse process. The thesis refines knowledge management concerning intangible reuse.

A new finding is that software asset reuse requires a proactive decision to anticipate a scarcity of knowledge in space or time, which has been identified as the software asset reuse trigger. Reuse sets in before the life end of the software asset is reached. It creates parallel software instances via abstraction, repurposing and adaptation. These boost the asset lifetime as they are logically linked. They represent tailored solutions for a heterogeneous client base and, therefore, target business-to-business niche markets.

This thesis makes an original contribution to knowledge by identifying that collaborative sharing of the change required for one client with existing reusers leads to improved software quality, surpassing that of other software constructs. Further, it claims that reusable software assets target a parallel market to software products.

This research significantly contributes to practice: First, by demonstrating that reuse is only feasible if the software asset can be adapted. Second, one reason for being of some reusable software assets is to stop the flood of less funded individual software trying to serve the same need. Third, a managerial guide provides advice on building reuse capabilities in the IT organisation to support the change that drives software asset reuse.

Acknowledgements

Aiming for a Doctor of Business Administration (DBA) degree was only possible with the help and encouragement of various people.

Many thanks go to my supervisors, Professor Paul Sudnik and Dr Jesus Canduela, for their guidance, professional support and the opportunity to research a subject I am passionate about.

I am grateful for my business network, particularly Ray Harishankar, Michael Martine and their Global Asset Team, who supported my research with expertise, mentoring and drafts review.

Finally, special thanks go to my husband, our three children and my parents, who supported my DBA journey continuously.

I dedicate this research to my 97-year-old grandfather, Dr Norbert Hellmann, the first in our family to achieve a doctoral degree.

Grandpa, your academic success has inspired me. Your wisdom and knowledge impacted the life of your children, grandchildren and great-grandchildren. I want to continue. Thank you for always being there. It means a lot to me.

I have enjoyed every moment I spent on this thesis. It is about sharing knowledge and saving resources. It will hopefully help make this world a better place.

Table of Contents

Abstract.....	i
Table of Contents	iii
Glossary of Terms.....	vi
List of Abbreviations	vii
List of Tables	viii
List of Figures.....	ix
1. Introduction.....	1
1.1 Research Area	1
1.2 The Problem Faced	3
1.3 Addressed Research Gaps and Motivations	3
1.4 Research Focus.....	6
1.5 The Contribution of this Research to Knowledge and Professional Practice ...	6
1.6 The Structure of this Work.....	9
2. Literature Review and Theoretical Framework	11
2.1 Software Asset Reuse IT Research to Date.....	11
2.2 Intellectual Capital	13
2.2.1 Types of Intellectual Capital.....	13
2.2.2 Intangible Assets and their Organisational Performance	14
2.2.3 Assetisation.....	20
2.2.4 Servitisation	23
2.3 Circular Economy	24
2.3.1 Reuse	25
2.3.2 Economy of Reuse.....	29
2.4 Knowledge Management	30
2.4.1 Types of Knowledge.....	30
2.4.2 Knowledge Content	33
2.4.3 Knowledge Work Context.....	35
2.4.4 Management of Knowledge Processes	37
2.4.5 Knowledge Creation	38
2.4.6 Knowledge Storage.....	40
2.4.7 Knowledge Distribution	43
2.4.8 Knowledge Application and Reuse	46
2.5 Business IT.....	48
2.5.1 Software and Services	48

2.5.2	Reusable Software Asset Creation	52
2.5.3	Reusable Software Asset Storage	54
2.5.4	Reusable Software Asset Distribution.....	55
2.5.5	Reusable Software Asset Reuse	57
2.5.6	The Impact of Reuse on Innovation	61
2.6	Objective of this Research and Research Questions	62
2.7	Theoretical Framework	63
3.	Research Methodology	66
3.1	Research Philosophy	66
3.2	Research Design.....	68
3.2.1	Research Approach.....	68
3.2.2	Research Strategy	68
3.2.3	Research Methodology	70
3.2.4	Time Horizon.....	71
3.2.5	Research Techniques for Data Collection and Analysis	71
3.3	Case Study Planning	73
3.3.1	Case Study Design Setup.....	73
3.3.2	Case Study Data Collection Procedure.....	75
3.3.3	Case Study Quality Assurance	77
3.3.4	Pilot Study	80
3.4	Ethics.....	81
4.	Case Study Presentation, Findings and Discussion	83
4.1	Case Study Presentation.....	83
4.1.1	Case Study Execution.....	83
4.1.2	Case Study Analysis	88
4.1.3	Research Diagram.....	92
4.2	Findings Case 1 – Reuse Initiated during Sales Generating Savings.....	94
4.2.1	Case 1 Asset Creation.....	94
4.2.2	Case 1 Asset Coding and Storage.....	96
4.2.3	Case 1 Asset Distribution	101
4.2.4	Case 1 Asset Reuse.....	106
4.3	Findings Case 2 – Reuse Initiated during Sales Generating Revenue	110
4.3.1	Case 2 Asset Creation.....	110
4.3.2	Case 2 Asset Coding and Storage.....	115
4.3.3	Case 2 Asset Distribution	122
4.3.4	Case 2 Asset Reuse.....	126
4.4	Findings Case 3 – Reuse Initiated during Solutioning Generating Revenue .	130

4.4.1	Case 3 Asset Creation.....	130
4.4.2	Case 3 Asset Coding and Storage.....	132
4.4.3	Case 3 Asset Distribution	139
4.4.4	Case 3 Asset Reuse.....	142
4.5	Findings Case 4 – Reuse Initiated during Solutioning Generating Savings ..	150
4.5.1	Case 4 Asset Creation.....	151
4.5.2	Case 4 Asset Coding and Storage.....	153
4.5.3	Case 4 Asset Distribution	156
4.5.4	Case 4 Asset Reuse.....	159
4.6	Findings Case 5 – Reuse Initiated during Delivery Generating Savings	166
4.6.1	Case 5 Asset Creation.....	166
4.6.2	Case 5 Asset Coding and Storage.....	169
4.6.3	Case 5 Asset Distribution	173
4.6.4	Case 5 Asset Reuse.....	176
4.7	Case Study Discussion	182
4.7.1	Asset Creation.....	182
4.7.2	Asset Coding and Storage.....	186
4.7.3	Asset Distribution.....	190
4.7.4	Asset Reuse.....	196
4.8	Answers to the Research Questions	204
5.	Contributions and Conclusion.....	223
5.1	Contribution to Knowledge.....	223
5.2	Contribution to Practice	227
5.3	Limitations	227
5.4	Recommendations for Future Research	228
5.5	Conclusion.....	230
	References.....	231
	Appendices.....	258
	Appendix A: Operational Knowledge Processes	259
	Appendix B: Semi-structured Interview Guide	262

Glossary of Terms

Cloud – The delivery of computing services remotely via the internet

DB2 – A set of relational database products

GitHub – A platform where software code is hosted for collaboration and version control

Kubernetes - An open source system developed by Google for managing container applications

On-prem – Software installed on-premises, i.e., on local IT infrastructure – not on the cloud

Open source – In IT this means software code can be used, modified and redistributed freely without charges

List of Abbreviations

Abbreviation	Meaning
AI	Artificial Intelligence
API	Application Programming Interface
AWS	Amazon Web Services – a cloud computing platform
B2B	Business-to-Business
B2C	Business-to-Consumer
COTS	Commercial-off-the-Shelf
CRM	Client Relationship Management
DBA	Doctorate in Business Administration
HR	Human Resources
IC	Intellectual Capital
IFRS	International Financial Reporting Standards
IP	Intellectual Property
IT	Information Technology
JTB	Justified True Believe
NIH	Not Invented Here
PoC	Proof of Concept, demonstrates feasibility of a concept
Rfi	Request for Information
RfP	Request for Proposal
RPA	Robotics Process Automation
SaaS	Software as a Service
SAP	The name of both a company and its software
SPOC	Single Point of Contact
SWOT	Strength, Weakness, Opportunity, and Threat
UFT	Unified Functional Testing
VBA	Visual Basis Analysis
VP	Vice President

List of Tables

Table 1.1 Selection of companies worldwide known for reusing software (source: author)	5
Table 2.1 Grid-based system for defining reproducibility, replicability and generalisability/reuse - adapted from (Cacioppo et al., 2015; Schloss, 2018)	26
Table 2.2 Summary of consumption versus reuse (source: author).....	28
Table 2.3 Overview of factors impacting knowledge creation (source: author)	40
Table 2.4 Overview of factors impacting knowledge sharing (source: author)	44
Table 3.1 Philosophical views on knowledge - adapted from (Lai, 1981; Wickramasinghe et al., 2005)	67
Table 3.2 Research methodologies - adapted from (MacDonald & Headlam, 2008)	69
Table 3.3 Types of reuse cases to be selected for this case study (source: author)	74
Table 4.1 Interviewee list (source: author)	86
Table 4.2 Overview on the five cases constituting this case study (source: author; numbers provided for 2021)	87
Table 4.3 Content asset versus methodology asset (source: author)	200
Table 4.4 Comparing the use of conventional one-off software, reusable software asset-and software product (source: author)	206
Table 4.5 A reusable software asset versus software product - adapted from (Cacioppo et al., 2015; Schloss, 2018)	213
Table 5.1 Overview of how knowledge processes were bundled (source: author)	260

List of Figures

Figure 1.1 Software asset reuse combines knowledge management, software assets and reuse (source: author).....	2
Figure 1.2 Luggage handling software created in the Netherlands for Schiphol Airport in Amsterdam and two of its Reuses – at the Heathrow Airport London and in Germany at Deutsche Post – adapted from (Geology.com, 2021).....	4
Figure 2.1 Intellectual capital - adapted from (Abd-Elrahman & Ahmed Kamal, 2020; Azzahra, 2018; Flores et al., 2020)	13
Figure 2.2 Relationship between human capital, relational capital and structural capital – adapted from (Agostini & Nosella, 2017).....	14
Figure 2.3 Types of assets by physical existence - adapted from (Mehta & Madhani, 2008)	15
Figure 2.4 Valuation of software assets - adapted from (Wilson et al., 2000)	17
Figure 2.5 Reuse benefit initiation along the software lifecycle (source: author)	18
Figure 2.6 Innovation and patent strength (Haskel & Westlake, 2022)	19
Figure 2.7 Reuse as part of the circular economy (Geissdoerfer, 2020; Press room European Commission, 2014)	25
Figure 2.8 A model of knowledge work - adapted from (Wijnhoven, 2008).....	31
Figure 2.9 Relationship between data, information and knowledge based on Russell Ackoff’s Apex - adapted from (Krishnaveni & Raja, 2012; Oppenheim et al., 2003).....	33
Figure 2.10 Difference between data, information, knowledge, insight and wisdom (MacLeod & Sommerville, 2016)	34
Figure 2.11 Knowledge transfer (Oppenheim et al., 2003)	35
Figure 2.12 Socialisation, externalisation, combination and internalisation (SECI) - adapted from (Dreyer & Wynn, 2016; Nakamori, 2020; Nonaka & Teece, 2001)	36
Figure 2.13 Individual and group tacit knowledge process – triggers are represented by a crossed-out circle (Dreyer & Wynn, 2016)	37
Figure 2.14 Relationship between explicit, tacit and implicit knowledge - adapted from (Cook & Brown, 1999; Nickols, 2000)	39

Figure 2.15 Conceptual knowledge storage/retrieval system success model (Taraszewski, 2017).....	42
Figure 2.16 Tailored one-off IT-service (I) vs reusable software asset (II) vs software product (III) - adapted from (Wolfram et al., 2020)	49
Figure 2.17 Four ways to obtain a reusable software asset for internal reuse - adapted from (Mateen et al., 2017).....	53
Figure 2.18 The sharing economy business model (Beha et al., 2022).....	56
Figure 2.19 Examples of reusable software-related artefacts - adapted from (Bauer, 2016)	57
Figure 2.20 Theoretical software asset reuse scenario (source: author).....	58
Figure 2.21 Considering an asset for reuse (source: author)	59
Figure 2.22 Intellectual capital directly impacts innovation - adapted from (Alrowwad & Abualoush, 2020).....	64
Figure 3.1 Saunders' research onion simplified - adapted from (Saunders et al., 2019, Fig. 4.1).....	66
Figure 3.2 CRM process stages - adapted from (Salesmate, 2018).....	74
Figure 4.1 Screenshot of NVivo analysis – identified codes appear in the left column (source: author)	89
Figure 4.2 Screenshot of Excel analysis – identified codes appear in the left column (source: author)	91
Figure 4.3 Summarised research approach (source: author).....	93
Figure 4.4 Case 1 - Location of asset owner versus asset reuser - adapted from (Geology.com, 2021)	94
Figure 4.5 Discussion topics of Section 4.2.1 (source: author).....	94
Figure 4.6 Discussion topics of Section 4.2.2 (source: author).....	97
Figure 4.7 Discussion topics of Section 4.2.3 (source: author).....	101
Figure 4.8 Discussion topics of Section 4.2.4 (source: author).....	106
Figure 4.9 Reuse and process timeline for reuse Case 1 (source: author).....	109
Figure 4.10 Case 2 - Location of asset owner versus asset reuser (adapted from (Geology.com, 2021)	110
Figure 4.11 Discussion topics of Section 4.3.1 (source: author).....	110
Figure 4.12 Discussion topics of Section 4.3.2 (source: author).....	115
Figure 4.13 Discussion topics of Section 4.3.3 (source: author).....	122

Figure 4.14 Discussion topics of Section 4.3.4 (source: author)	126
Figure 4.15 Reuse and process timeline for reuse Case 2 (source: author)	129
Figure 4.16 Case 3 - Location of asset owners versus asset reuser - adapted from (Geology.com, 2021)	130
Figure 4.17 Discussion topics of Section 4.4.1 (source: author)	130
Figure 4.18 Discussion topics of Section 4.4.2 (source: author)	133
Figure 4.19 Discussion topics of Section 4.4.3 (source: author)	139
Figure 4.20 Discussion topics of Section 4.4.4 (source: author)	143
Figure 4.21 Reuse and process timeline for reuse Case 3 (source: author)	149
Figure 4.22 Case 4 - Location of asset owner versus asset reuser - adapted from (Geology.com, 2021)	150
Figure 4.23 Discussion topics of Section 4.5.1 (source: author)	151
Figure 4.24 Discussion topics of Section 4.5.2 (source: author)	154
Figure 4.25 Discussion topics of Section 4.5.3 (source: author)	157
Figure 4.26 Discussion topics of Section 4.5.4 (source: author)	159
Figure 4.27 Reuse and process timeline for reuse Case 4 (source: author)	165
Figure 4.28 Case 5 - Location of asset owner versus asset reuser - adapted from (Geology.com, 2021)	166
Figure 4.29 Discussion topics of Section 4.6.1 (source: author)	167
Figure 4.30 Discussion topics of Section 4.6.2 (source: author)	169
Figure 4.31 Discussion topics of Section 4.6.3 (source: author)	174
Figure 4.32 Discussion topics of Section 4.6.4 (source: author)	176
Figure 4.33 Reuse and process timeline for reuse Case 5 (source: author)	181
Figure 4.34 Discussion topics of Section 4.7.1 (source: author)	182
Figure 4.35 Summary of case study outcome – asset creation (source: author)	186
Figure 4.36 Discussion topics of Section 4.7.2 (source: author)	186
Figure 4.37 Summary of case study outcome – asset coding and storage (source: author)	190
Figure 4.38 Discussion points of Section 4.7.3 (source: author)	190
Figure 4.39 Reuse as part of the circular economy - adapted from (Stahel, 2016)	195

Figure 4.40 Summary of case study outcome – asset distribution (source: author).....	196
Figure 4.41 Discussion topics of Section 4.7.4 (source: author).....	196
Figure 4.42 Benefits of software asset reuse – source (Cases 1–5 – bold entries) and (Barros-Justo et al., 2018; Horne, 1995; Majchrzak et al., 2004; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Salgot et al., 2017; Sandhu & Batth, 2021b; Wilson et al., 2000).....	201
Figure 4.43 Summary of case study outcome – asset reuse (source: author).....	201
Figure 4.44 Summary of the key case study findings (source: author).....	203
Figure 4.45 Characteristics of the reusable software asset (source: author).....	205
Figure 4.46 Reusable software asset versus software product - adapted from (Wolfram et al., 2020).....	207
Figure 4.47 Peer clients co-shape the reusable software asset (source: author).....	207
Figure 4.48 The reusable software asset is a specific software asset - adapted from (Mehta & Madhani, 2008).....	208
Figure 4.49 The reusable software asset positioned against the software product (source: author).....	209
Figure 4.50 Software asset reuse process (source: author).....	211
Figure 4.51 Marrying a reusable software asset to an IT business project (source: author).....	212
Figure 4.52 Software asset reuse according to Cases 1–5 of the Case Study (source: author).....	214
Figure 4.53 Teams involved in software asset reuse (source: author).....	217
Figure 4.54 Overview of Marketing Methods for Reusable Software Assets (source: author).....	219
Figure 4.55 Software asset reuse tasks (source: author).....	221
Figure 4.56 Practical implications of software asset reuse (source: author).....	222

1. Introduction

This Chapter introduces the research area and problem faced in Sections 1.1 and 1.2, respectively. It describes the research gap (Section 1.3) and the subsequent research focus (Section 1.4). Finally, it previews the research contributions in Section 1.5 before it details the structure of this work (Section 1.6).

1.1 Research Area

Experts worldwide have recognised the significant technical and business potential of reusing software code and its related artefacts (Griss, 1993). McIlroy (1968) first suggested using the software code from one commercial project in another. Single reuse can comprise a complex IT solution worth several million dollars. Any reuse improvements help global IT enterprises increase revenue from their multi-million-dollar software asset licence and related services. “The USA department of defense saves \$300 million yearly by increasing the level of reuse only 1%” (AL-Badareen, 2021, p. 18). Additionally, increases in productivity, quality, performance, timeliness, innovation and portability, and a reduction in cost, waste, and testing, are just a few of the reuse benefits (Barros-Justo et al., 2018). Furthermore, software reuse is of academic interest since its principles are relevant to related areas such as automation, cloud software or other intangible reuse like media asset reuse.

Reuse benefits from the intellectual progress made by previous knowledge workers (Vom Brocke et al., 2015). This heightened knowledge level increases innovation chances (Majchrzak et al., 2004; Ritter, 1838; Schumpeter, 2017).

Bernard et al. (2014) recommend looking at successful examples to draw comparisons to a given situation. This can lead to new approaches – one of the intentions of this research. Figure 1.1 forms the foundation of this thesis by splitting the required research into three dedicated domains on top of IT: software, assets and reuse.

Software is a machine-readable form of explicit business and technical knowledge (Di Cosmo & Zacchiroli, 2017; Dreyer & Wynn, 2016; Huang, 2019; Kneuper, 2002). Software asset management can be regarded as a special kind of well-researched knowledge management. This research, therefore, focuses on *knowledge management*. Alavi & Leidner (2001), along with many others, have posited a structure for knowledge management whereby knowledge can be created, stored, shared, and reused.

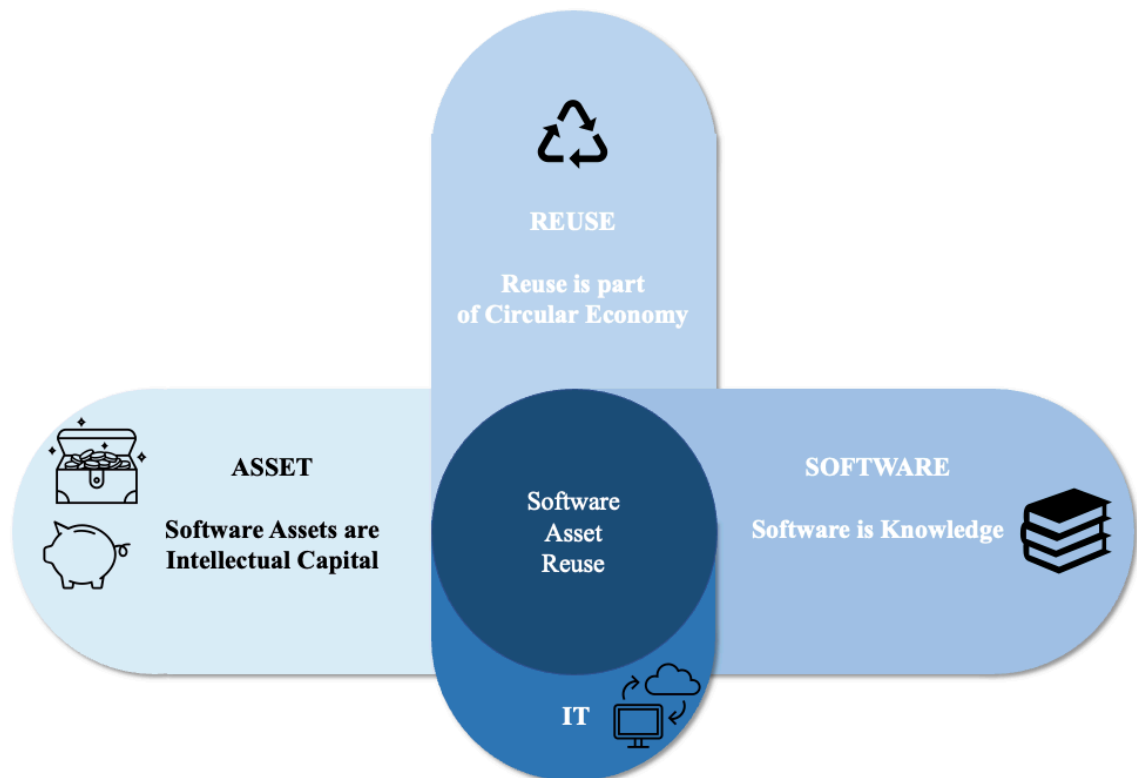


Figure 1.1 Software asset reuse combines knowledge management, software assets and reuse (source: author)

At the same time, software has a financial value (Horne, 1995). It is a special kind of asset – an *intangible asset*. Birch (2020) researched the theory of assets in general, and Haskel & Westlake (2018) studied the behaviour of intangible assets. Intangible assets represent intellectual property (IP), a special kind of intellectual capital (IC). Therefore, this research goes beyond the scope of IT because a part of the literature review details the characteristics of intellectual capital and intangible assets.

As the name implies, reusable software assets can be reused. Early software reuse research was IT-driven. However, *reuse* is more than that. Succar (2013) discusses reuse in the building industry in a way relevant to this research. Reuse is a crucial element in the circular economy (Stahel & MacArthur, 2019).

Subsequently, *intellectual capital*, *circular economy* and *knowledge management* are identified as research domains and considered first in the literature review and later in the case study. The answers to the research questions contribute to these three domains by advancing the academic literature in these fields.

1.2 The Problem Faced

In practice, software asset reuse is cumbersome to achieve and problematic to measure (Corrado et al., 2021; Goodridge et al., 2016; Mäkitalo et al., 2020). IT providers anticipated substantial financial benefits and increased productivity and quality (Fadler & Legner, 2020; Frakes & Terry, 1996). However, the reality has never matched these expectations, as reuse has only happened on an ad hoc basis (Maccanti et al., 2016).

Researchers and practitioners have no single approach to address this issue (Enders et al., 2016). Terjesen (2003), for example, provides a symptom description of the obstacles to knowledge reuse at Accenture – but no solution. Her report supports Griss' (2015) conclusion that reusing software assets is challenging.

1.3 Addressed Research Gaps and Motivations

This thesis contributes to solving the identified problem by addressing three gaps in the research. First, previous researchers (Bauer, 2016; Spoelstra et al., 2011) interviewed reuse experts on their broad reuse experience without referring to particular reuse cases. Keswani et al. (2014) and Brown et al. (2021) criticise the literature's lack of real-life reuse success stories. Current literature typically presents university-constructed reuse examples. The few real-life reuses are presented by managers who were not hands-on involved or experts who talk generally about reuse instead of specific reuse details. Further, reuse is presented without highlighting required sales/distribution processes that are required to make reuse a valid option to use. Advancing circular economy theory, this research addresses the literature gap for the first time by presenting and qualitatively analysing five successful, recent, *real-life reuses of software assets* established in the market.

Second, this research targets Bauer's (2016, p. 4) point that reuse "requires a global focus of management transcending the scope of single projects". Although reuse faces technological issues, it remains a managerial and organisational challenge (Cusumano, 1991; Shiva & Shala, 2007; Zhou, 2001). It is unclear what differentiates a reusable software asset from a software product and when and where it can be applied in business. Little is known about the characteristics of reusable software assets in terms of the longevity of the reusable items or related processes such as assetisation and servitisation. This research addresses the *business aspects of software asset reuse* by advancing academic literature on intellectual capital.

Third, Barros-Justo et al. (2019, p. 15) explicitly list in their tertiary study the need to “deepen the knowledge about the *processes of reuse* and how the benefits are transferred to the industry”. Current software asset reuse literature fails to describe what triggers reuse. The reuse process's superior impact on software quality is not detailed enough. The role of processes, such as adaptation, abstraction, repurposing, etc., must be better explained along with their impact on innovation. So far, the research draws from ad hoc reuse, leaving it unclear how systematic reuse can be achieved. This thesis expands the knowledge management theory by investigating the knowledge processes of software asset reuse.



Figure 1.2 Luggage handling software created in the Netherlands for Schiphol Airport in Amsterdam and two of its Reuses – at the Heathrow Airport London and in Germany at Deutsche Post – adapted from (Geology.com, 2021)

Beyond academic relevance, addressing these three research gaps has practical business relevance. Figure 1.2 shows an illustrative real-life B2B reuse example of a software asset. The luggage handling software asset created for Schiphol Airport in Amsterdam (Newsroom, 2009) doubled the baggage handling capacity and tracked each bag using robot-handling software (Airport-Industry-Review-cms, 2011). Later, it was slightly modified and reused at Terminal 5 of London Heathrow Airport (Okosun, 2011). When the need for brand-new baggage routing software dropped off, the use case was generalised and expanded from baggage handling to material flow so that the software

asset could be reused a second time to route parcels at Deutsche Post in Germany (Newsroom, 2017).

Along the lines of this example, mainly unnoticed by the public, the reusable software asset has emerged – as an alternative to the prominent software product which cannot be reused. Table 1.1 lists a selection of companies known for their software reuse initiatives. This testifies the wide practical relevance of software asset reuse.

Table 1.1 Selection of companies worldwide known for reusing software (source: author)

Companies engaged in software asset reuse – in alphabetical order	
Accenture (Terjesen, 2003)	Motorola (Barros-Justo et al., 2018; Frakes et al., 1991; Irshad, 2010)
AT&T (Frakes et al., 1991)	NASA (Irshad, 2010; Lange, 2020)
Boeing (Frakes et al., 1991)	NATO (McIlroy et al., 1968)
Contel (Prieto-Díaz, 1991a)	NEC (Cusumano, 1991; Griss, 2015)
Ericsson (Griss, 2015)	NTT DATA (Yoshida, 2014)
European Space Agency (Lange, 2020)	Orbotech (Irshad, 2010)
Freescale Semiconductor India Pvt Ltd. (Barros-Justo et al., 2018)	Statoil KTJ/IT (Barros-Justo et al., 2018)
Fujitsu (Barros-Justo et al., 2018; Cusumano, 1991)	STZ Gesellschaft für Softwaretechnologie (Frakes et al., 1991)
Google (Bauer & Vetro, 2016)	Tata Consulting Services (Shiva & Shala, 2007)
Hitachi (Barros-Justo et al., 2018; Cusumano, 1991)	Toshiba (Cusumano, 1991; Griss, 2015)
Hewlett-Packard (Griss & Wosser, 1995; Irshad, 2010; Shiva & Shala, 2007; Wentzel, 1994)	USA Department of Defense (AL-Badareen, 2021).
IBM (Keswani et al., 2014; Wolfram et al., 2020)	Volvo (Griss, 2015)

Asset reuse can generate a constant revenue stream, contributing millions of dollars to companies running a reuse program. Additionally, there are uncounted savings, efficiency effects and other reuse advantages like sustainability.

Reuse benefits can be gained both in IT as well as other industries. Examples could include component reuse during the design of space crafts (Lange, 2020) or agriculture products (Liu et al., 2020), production assets in filmmaking (Lear, 2021; Trottnow et al., 2020), building reuse (Fivet & Brütting, 2020), digital assets in construction (Succar & Poirier, 2020), stamping tool design knowledge reuse in sheet metal production (Jonsson et al., 2021) or reuse during the diagnostic process in the medical field (Shen & Spruit, 2019; Sridhar et al., 2012). They all deal with reusing knowledge assets to achieve innovation (Schumpeter, 2017). Knowing what processes are critical for systematic reuse would thus strengthen their position.

In summary, the research motivation is threefold. First of all, the identified research gaps will be addressed both from an academic and practical perspective. Second, many IT companies could practically benefit from the research results. Finally, new research could help make reuse more popular in general.

1.4 Research Focus

This thesis aims to answer three research questions based on empirical findings.

RQ1: How can a reusable software asset be characterised? – What is a reusable software asset? This question relates to the intellectual capital research domain and is of general research interest (Zabardast et al., 2022). It targets the research gap on the business aspect of software asset reuse identified in the literature by looking at the kind and characteristics of intellectual capital required for software asset reuse from a business perspective.

RQ2: How can the systematic reuse of a software asset in a global IT enterprise be specified? – What is systematic reuse? This question addresses the academic field of circular economy. It relates to the identified academic gap in real-life reuses of software assets.

RQ3: How should the knowledge-related processes, critical to enabling systematic software asset reuse in a global IT enterprise, be shaped? – How can systematic reuse be achieved? The answers to RQ1 and RQ2 are required to answer RQ3 – the central question in this research. RQ3 describes how to achieve the objective identified in RQ2 with the help of the reusable software asset characterised in RQ1.

The answer to RQ3 relates to the research domain knowledge management. It addresses the research gap in the field of knowledge management by detailing knowledge management processes, strategies and methodologies successful IT organisations must employ towards systematic software asset reuse.

1.5 The Contribution of this Research to Knowledge and Professional Practice

This research presents five recent, real-life, purposively selected software asset reuses to demonstrate the full spectrum of complex software asset reuse. The results gained are particularly valuable because the interviewed asset owners draw from a long reuse history with their respective reusable software assets. This study describes how the goal of systematic reuse can be achieved. On the other hand, it demonstrates the complexity of

the required knowledge management processes and thus implicitly reveals why many organisations fail.

This thesis contributes to the academic literature in the domain of *intellectual capital*:

Applying the theories of Abeysekera (2021, p. 1) on “knowledge-based value creation” and the theory of IFRS (2022) on software licensing helps *to characterise the reusable software asset in contrast to the software product*. The theoretical implication is to expand the theory of the International Financial Reporting Standards (2022) so that they explicitly differentiate between the two software assets – the reusable software asset and the software product. Consequently, this thesis helps bridge the discerned gap in the literature concerning the business dimension of software asset reuse, by positioning the reusable software asset as a viable counterpart to the more prevalently recognised software product. (Cusumano, 1991; Shiva & Shala, 2007; Zhou, 2001).

Adopting the theory of IFRS (2022) on the identified business need for reusable software assets leads to the theoretical implication that software asset reuse is *only applicable in B2B niche markets*.

The theory of Lacy et al. (2020) on how reuse expands the life of tangible items has the theoretical implication that it, even more, applies to intangibles due to the identified setting of *parallel intangible instances of reuse extending the longevity of the reusable items*.

Applying the theory of Romer (1990) and Haskel & Westlake (2018), assets can cause market concentration and dominance, of Birch & Muniesa (2020), assets often involve forms of monopoly control, and of Haskel & Westlake (2018), assets can have spillover, has the theoretical implication that the case study shows the spillover effect typically prevents monopolistic competition when intangibles such as software assets are reused.

This thesis adds to the academic literature in the domain of *circular economy*:

Responding to Keswani’s (2014) and Brown’s (2021) concern about not having enough *real-life software reuse examples* in the academic literature, the theoretical implications

of this case study first time describe *parallel intangible reuse instances* based on shared usership but kept ownership. A further implication is *reuse is a business*. Conducting an in-depth examination and analysis of five authentic software asset reuse cases through interviews with directly engaged individuals possessing extensive reuse experience contributes significantly to addressing the gap in the relevant scholarly literature Keswani et al. (2014) and Brown et al. (2021) mentioned.

Applying Griss' (1996) theory, the *basic characteristics of a reusable software asset* can be explained. The theoretical implications of the case study require a more comprehensive definition that includes the element of distribution to make intangible reuse happen on a large scale.

Following Wallace' (2018) theory, assets need to be pushed for reuse, which can either be horizontal or vertical (Anasuodei & Ojekudo, 2021; Jalender et al., 2010). The theoretical implications from this case study research *leverage the concept of vertical and horizontal reuse to the circular economy research* domain and add to the existing theory that at least *horizontal reuse requires a two-step sales/distribution* process.

This thesis contributes to the academic literature in the *knowledge management domain*:

According to Fogg's (2009) theory, motivation and ability are prerequisites. But it takes a trigger (Dreyer & Wynn, 2016) to change behaviour from use to reuse. This trigger is the scarcity (Salgot et al., 2017). The subsequent theoretical implication is that *the reuse of reusable software assets is triggered by knowledge scarcity*.

Applying Lacy's (2020) servitisation theory and Fayyaz' (2020) knowledge-sharing theory to the findings of this case study, the theoretical implications indicate *sharing the asset adaptations made for one client with the existing set of clients* establishes a feedback loop that provides the reusable software asset with a value higher than those of a software product.

Mapping Wegener's (2016) creativity and knowledge brokering theory and Sandhu & Batth's (2021a) abstraction theory against this research leads to the theoretical implication that *abstraction is key for knowledge generation* and for virtually linking individual knowledge instances.

Finally, applying Alrowwad & Abualoush's (2020) theory, according to which knowledge leads to innovation, has the theoretical implication that *innovation can be achieved by abstraction and reuse of knowledge assets* in niche areas. This contribution along with the other findings of this thesis, help close the literature gap on the knowledge process details relevant to software asset reuse Barros-Justo et al. (2019, p. 15) have described.

This thesis contributes to practice by demonstrating that reuse is based on the adaption of the software asset. Second, it identifies the right of existence for some reusable software assets with the need to stop the flood of poorly funded software, all serving the same purpose. Third, this research provides a management guide to establish the organisational capabilities that enable systematic software asset reuse in a global IT enterprise and support collaborative software change, identified as the driving reuse factor.

Systematic reuse of knowledge can change our way of working in political-economic terms. It increases our ability to solve local and global problems.

1.6 The Structure of this Work

The following research work has two main parts: theoretical in Chapter 2 and practical in Chapter 4.

The literature review in Chapter 2 provides theoretical research by covering the four wider research areas identified in Figure 1.1 - intellectual capital (2.2), circular economy (2.3), knowledge management (2.4), and IT (2.1 and 2.5). Based on the existing literature, this helps to describe what a reusable software asset (RQ1) and what reuse (RQ2) is. Eventually, Chapter 2 concludes by detailing the theoretical framework that forms the basis of this research.

The chosen research methodology is explained in Chapter 3 by detailing the philosophical paradigm, research design, case study planning and research ethics relevant for this research.

Chapter 4 lists the specifics of applying this research methodology to this thesis. It presents practical research via a case study. It focuses on four key knowledge management processes resulting from the exercise in "Appendix A: Operational

Knowledge Processes” and culminates in answering the research questions. The case study and the literature research results enable responses to the research questions. In addition, this chapter critically reviews literature findings in practice.

Finally, Chapter 5 summarises the contributions to knowledge and practice this thesis claims. It states research limitations and provides recommendations for future research.

2. Literature Review and Theoretical Framework

Figure 1.1 structures Chapter 2 and provides initial, literature-based answers to the set research questions. Past researchers have addressed the stagnating, non-systematic reuse of software by proposing *IT* solutions (Section 2.1). Taking a business view, Section 2.2 introduces software assets as *intellectual capital* and provides RQ1-relevant insights into assets, in general, and software assets, in particular. Section 2.3 presents reuse RQ2-relevant as part of the *Circular Economy*. Concerning RQ3, Section 2.4 dives deeper by giving an overview of knowledge, *knowledge management* processes and enablers.

Closing the loop in Figure 1.1, Section 2.5 describes *IT* software services. It explores why the concept of an asset is necessary for software reuse. Eventually, Section 2.6 specifies the research objective and research questions of this thesis. Finally, Chapter 2 concludes with Section 2.7, detailing the theoretical framework that forms the basis for this entire piece of research.

2.1 Software Asset Reuse IT Research to Date

The IT aspects of software asset reuse have been widely discussed in the literature. Davis (1992) and Frakes (1986) are key early researchers. Additionally, Rubén Prieto-Díaz's publications symptomatically reflect the history of software reuse. They cover topics like software classification (Prieto-Díaz, 1985; Prieto-Díaz & Freeman, 1987), building and managing software libraries (Jones & Prieto-Díaz, 1988), domain analysis (Prieto-Díaz, 1990), systematic classification using semantic knowledge categories (Prieto-Díaz, 1991a, 1991b), practical reuse experiences (Frakes et al., 1991), an AI-based library system for software reuse (Ostertag et al., 1991), domain-relevant ontologies (Prieto-Díaz, 2003), requirements (Prieto-Díaz, 2004) and use cases (Génova et al., 2005). The paper "Reuse is dead, long live reuse" (Zand et al., 2001) marks the end of what could be referred to as early reuse euphoria (Griss, 2015; Zand et al., 1999).⁴

Despite the setbacks, however, software asset reuse was not abandoned by researchers worldwide. For example, Brazilian Vinicius Cardoso Garcia looks into reuse process maturity levels (2007) and reuse capability levels (2008). He concludes that "the absence of a clear reusability strategy and the lack of specific top-management support" (Garcia et al., 2007, p. 61) could explain the stagnation in the adoption of software reuse. Later Spoelstra et al. (2011, p. 1), in the Netherlands, point out that "software reuse has never

reached its full potential” because “organisations are not aware of the different levels of reuse or do not know how to address reuse issues”.

Yoshida (2014) explains how NTT DATA in Japan digitises software assets to make them available for computer processing to support software asset distribution. Bauer (2016, p. 17) concludes from her case study in Germany that “successfully introducing structured reuse ... is a non-trivial task that transcends typical organisational boundaries, e.g. of departments, and units of work planning, such as single projects and, thus, requires a holistic view of the organisation”. Keswani et al. (2014) from India list predominantly non-technical barriers to software asset reuse, such as organisational inability to match the existing software with any new requirements, economic barriers to supporting enterprise-wide reuse investment, administrative impediments leveraging reuse across organisational and geographic boundaries. After conducting quantitative research in Morocco, Younoussi (2019, p. 165) confirms that “the reuse success in organizations is due to individual efforts and not to organizational process”.

Many assumptions exist about why software asset reuse has stagnated, but no alternatives have been offered. This may, in part, be because the relevant researchers did not look beyond their relatively narrow IT domain.

Before answering RQ3 about how systematic software asset reuse can be achieved in daily business, it is essential to define the terms reusable software asset for RQ1 and systematic reuse for RQ2.

Zhou (2001) and Spoelstra (2011) provide the most detailed explanations of a reusable software asset to date. They perceive a reusable software asset as a bundle of reusable software code and related artefacts like test cases, functional designs, code components, test scenarios, objects, design models, domain architecture, database schema, documentation, manuals, standards and sales material but do not separate it from the software product. They miss to explain the characteristics that make a bundle of software reusable.

Based on current software asset reuse literature, Griss (1996, p. 1) provides the most advanced answer to RQ2 by stating systematic reuse is an “institutionalised organisational approach to ... development in which reusable assets are purposely created or acquired and then consistently used and maintained to obtain high levels of reuse,

thereby optimising the organisation’s ability to produce quality software ... rapidly and effectively.” Merriam-Webster (2021b) defines systematic as “methodological in procedure or plan”. However, even the combined definitions do not prove to be sufficient in practice. They are limited to the software's technical creation, maintenance and use and miss addressing the distribution aspects of software asset reuse.

A comprehensive definition of systematic software asset reuse should clarify what triggers software reuse and detail how reuse is organisationally implemented from a business point of view, not just IT. Further, it should list prerequisites and link back to the theory of reuse, bearing in mind the specifics of intangibles.

2.2 Intellectual Capital

The previous section described the IT aspects of software asset reuse. This section progresses in covering the aspects shown in Figure 1.1. It classifies software assets as intellectual capital and discusses their organisational performance.

2.2.1 Types of Intellectual Capital

Intellectual capital covers the three core competencies: human, relational, and structural capital – see Figure 2.1. According to Agostini & Nosella (2017), *human capital* impacts relational and structural capital.

Intellectual Capital

Human capital

- Education/training
- Competences
- Wellbeing
- Innovation

Relational Capital

- Relationships with customers, suppliers
- Trade names
- Licences
- Franchises
- Customer knowledge
- Marketing capabilities

Structural Capital

- Organisational capital
 - Organisation philosophy
- Process capital
 - Techniques
 - Procedures
 - Projects and initiatives
- Innovation capital
 - Non-physical infrastructure
 - Intellectual property
 - Patents
 - Trademarks
 - Copyrights
 - Intangible assets
 - Software
 - Databases

Figure 2.1 Intellectual capital - adapted from (Abd-Elrahman & Ahmed Kamal, 2020; Azzahra, 2018; Flores et al., 2020)

As seen in the *structural capital column* of Figure 2.1, intellectual property is part of a company’s innovation capital. Software is an intangible asset, “a resource that is non-physical, non-financial, has [a] long life, and has potential to provide future benefits to

the owner.” (Park et al., 2023, p. 1). Besides patents, trademarks, copyrights and others, it forms a company's intellectual property.

Human capital, for example, in the form of well-trained innovative experts, can create *relational capital*. People can partner with customers, suppliers or peer providers (Ahmed et al., 2021), resulting in licences or other agreements.

Human capital, relational capital and structural capital can enable radical innovation. Figure 2.2 shows human capital as the root cause. The right staff, with all their skills, belief, and persistence, is essential for business innovation.

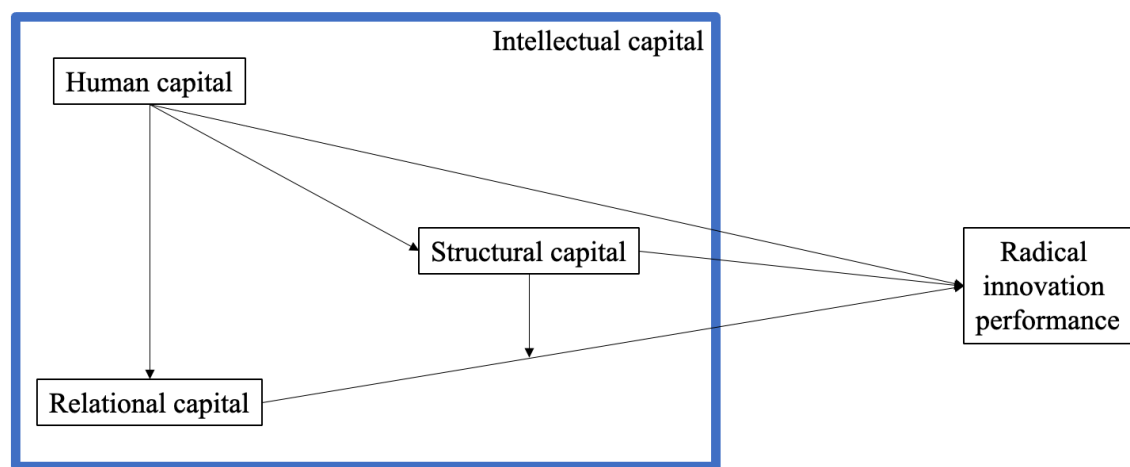


Figure 2.2 Relationship between human capital, relational capital and structural capital – adapted from (Agostini & Nosella, 2017)

Global IT enterprises generate revenue and profit from their software and related services. They use intangible intellectual capital as input for “knowledge-based value creation” (Abeysekera, 2021, p. 1). IT companies must manage, measure, report and strategically align it to secure market advantage (Abualoush et al., 2018; Khavandkar et al., 2016). Their output, software and related services, is competitive if it has been created by particularly selected, talented experts (Ahmed et al., 2021).

2.2.2 Intangible Assets and their Organisational Performance

The last section illuminated the different kinds of intellectual capital. It underscored the influence of intangible assets, specifically intellectual capital, within contemporary organisations by driving radical innovation performance.

This new section delves into the realm of intangible assets by focusing on one of the most vital categories within this domain: computer software. It will discuss how the reuse of

software assets drives innovation, efficiency, and competitiveness in the modern business landscape - considering both revenue and cost-saving aspects.

The section recognises the unique economic characteristics such as scalability, spillover effects, sunk costs, and synergies that define intangible assets in general and software assets in particular.

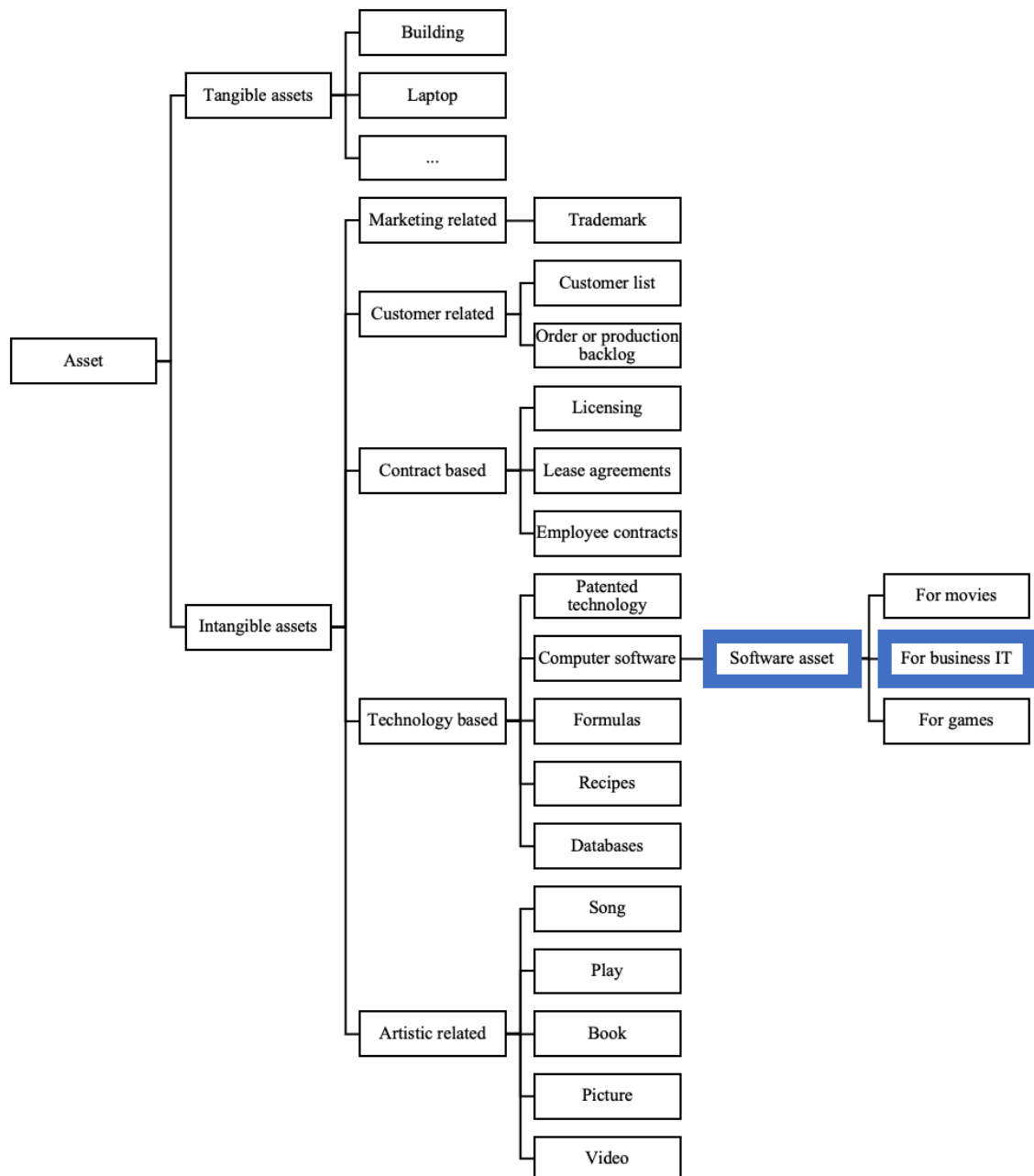


Figure 2.3 Types of assets by physical existence - adapted from (Mehta & Madhani, 2008)

Derived from the French word *assez*, enough, an asset is something precious, “any component, model, process or framework of value that can be leveraged or reused.” (Definitions.net, 2022). Assets can be characterised according to their physical existence (Figure 2.3).

Assets can be tangible or intangible. Software assets fall into the category of intangible *assets*. “The subject *intangible* means it does not have physical substance, but grants rights of ownership to the owner and economic benefits” (Milala et al., 2021, pp. 2555 - 2556).

Over the past century, intangibles have started to play a steadily more important role in economic growth (Fu & Ghauri, 2021; Haskel & Westlake, 2018; Lev, 2018). *Technology-based intangible assets* enable organisations to innovate processes, increase efficiency and improve the quality of services and products.

One specific *technology-based* intangible asset is computer software as defined by the latest International Financial Reporting Standards (IFRS) definition: “An intangible asset is an *identifiable* non-monetary asset without physical substance. Such an asset is identifiable when it is separable, or when it arises from contractual or other legal rights. Separable assets can be sold, transferred, licensed, etc. Examples of intangible assets include computer software ...” (IFRS, 2022, p. 1).

The value of intangible assets impacts *organisational competitiveness* (Abualoush et al., 2018) as they enable organisations to innovate, improve efficiency and help to differentiate from competition. This is in line with Agostini & Nosella’s (2017) statement on the innovative impact of intellectual capital (Figure 2.2). It gains additional importance as “the economy is inverting from one where value was measured by ‘touch’ to one where value is driven by thought. This change has been no less significant than the industrial revolution more than a century ago” (Tomo, 2020, p. 3).

One attribute of software reuse is its influence on organisational performance. To gain a comprehensive understanding of this relationship, it is important to consider various organisational performance theories that can complement each other in explaining the underlying mechanisms. For instance, as refined by Barney.

Field (2001), the resource-based view highlights the role of unique and valuable resources in driving superior performance, which can be linked to the efficient utilisation of reusable software assets.

Becker's human capital theory (England & Folbre, 2023) can be connected to software reuse by underlining the importance of employee development and expertise in effectively repurposing software assets. Additionally, Teece's dynamic capabilities framework (Marrucci et al., 2022) emphasises an organisation's ability to learn and adapt under pressure, which can be associated with the agility required to identify and exploit opportunities for software reuse.

Furthermore, Locke's goal-setting theory (Konstantara & Galanakis, 2022) can be applied to the context of software reuse by identifying the need for setting challenging objectives that encourage innovation and efficient resource utilisation. Lastly, Vroom's expectancy theory (Lloyd & Mertens, 2018) can provide insights into how the effort performance of outcome-motivated employees can contribute to the successful implementation of software reuse practices within an organisation.

When integrated and applied to the software reuse context, organisational performance theories offer a more comprehensive understanding of the relationship between software reuse and organisational performance. By acknowledging the complementary nature of these theories, this research aims to develop a more nuanced perspective on the impact of software reuse on organisational performance.

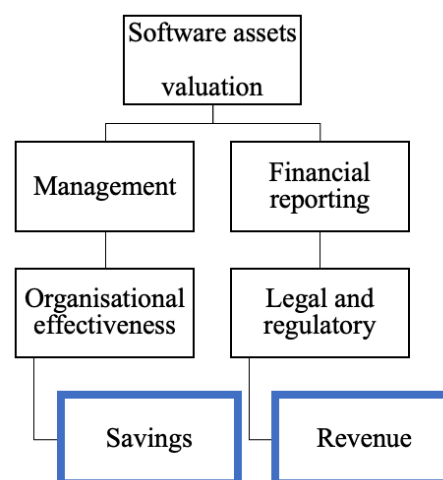


Figure 2.4 Valuation of software assets - adapted from (Wilson et al., 2000)

Savings and revenue are two key financial indicators that describe an organisation's performance (see Figure 2.4). *Revenue* represents an organisation's income. It typically correlates with the organisation's performance. Since software asset reuse revenue is typically received during operations, it must be agreed upon before contract closure (see Figure 2.5). If any reuse potential is identified after contract closure, it is too late contractually to agree on asset reuse revenue.

Savings represent a cost reduction and indicate an organisation's efficiency (see Figure 2.4). Savings can be initiated at any point in time and in parallel with any agreed revenue (see Figure 2.5). They are diminished by a certain amount of reuse cost that is an apportionment of the overall cost of managing the reuse programme (Biggerstaff, 1994).

Since reuse savings relate to benefits like the increase in quality, productivity, efficiency, maintainability, portability, knowledge sharing, customer satisfaction, product safety and the reduction of development costs and time, time to market and test time, they are difficult to track and specify (Irshad et al., 2016). While many publications and IT companies report these savings (Barros-Justo et al., 2018), specific figures are rarely provided (Svahnberg & Gorschek, 2017). There is general agreement that reuse in the first phases of a project results in greater benefits (Irshad et al., 2018).

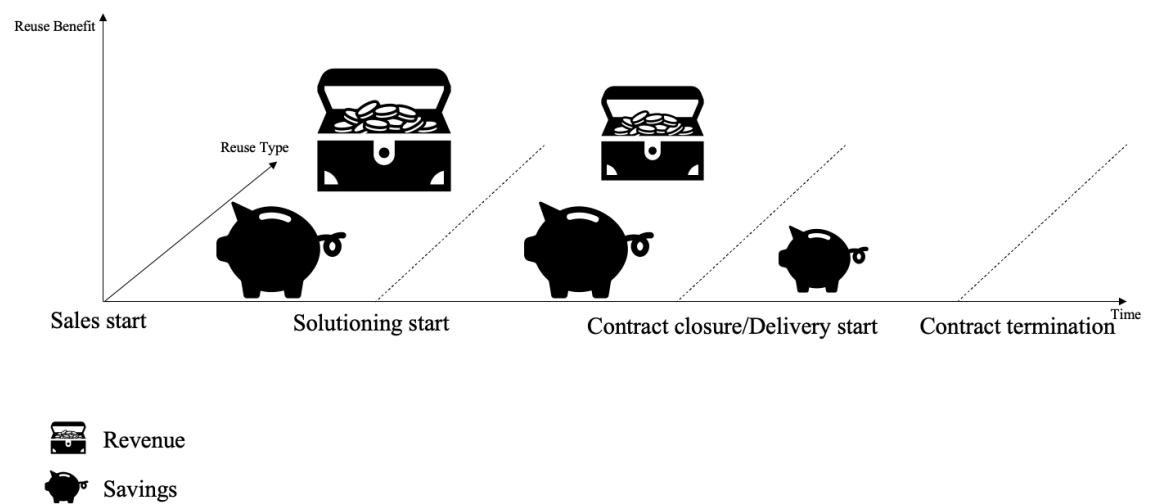


Figure 2.5 Reuse benefit initiation along the software lifecycle (source: author)

Software is knowledge and, therefore, beyond account measures. “Since knowledge is intangible, it is not measurable in accounting terms. In economics, knowledge is a factor of production; but in accounting, it is an owner’s capital” (Abeysekera, 2021, p. 1). Hence, some researchers recommend tracking innovative activities (Corrado et al., 2021) (Corrado et al., 2021), such as the number of patents.

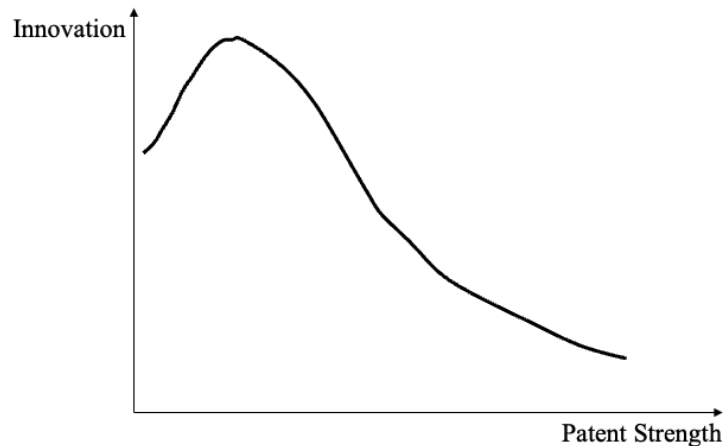


Figure 2.6 Innovation and patent strength (Haskel & Westlake, 2022)

Within limits, the importance of patents correlates with the importance of intangibles. Patents grant exclusive rights over a newly invented item or process representing a novel technical solution or approach to a problem (WIPO, 2022). Other than Corrado (2021), Haskel & Westlake (2022) see a patent strength dilemma (Figure 2.6). The number of patents does not rise with innovation but can go down. This dilemma is caused by the permanent trade-off between spillover effects and synergies of intangibles. It restricts the use of patent count to measure the value of intangible assets.

Intangibles, including software assets, show unusual economic characteristics, which Haskel & Westlake (2018) name the four S’s. First, an intangible asset is *scalable*, i.e. it can be used “again and again at relatively little cost, compared to most physical assets” (Haskel & Westlake, 2018, p. 65). Scalability is based on knowledge being scalable and an idea being non-exhaustive. In line with others, Haskel & Westlake (2018) see intangibles as non-rival. Consuming intangibles does not reduce the amount available – so they can be used by one client and parallel licensed to others at the same time (Jones & Tonetti, 2020) without price-taking competition. In conclusion, a specific intangible business concept emerges as the reuse of intangibles increases the return with each further

reuse (Mehta & Madhani, 2008). Haskel and Westlake (2018) foresee that intangible-intensive businesses can grow extensively, cause market concentration and clear market leader dominance – confirming Romer’s (1990) monopolistic competition.

Second, the costs of intangible assets are often *sunk* – they cannot be recovered if the business decides to back out. Dixit & Pindyck (1995, p. 636) call this value the “option value of a project resulting from its interdependence with future and follow-up investments”. Other researchers added two further financial aspects: The investment and capital in intangible assets is hard to measure from an accounting point of view, especially in global IT companies. Further, earnings, such as revenue, reflect the value of intangible assets better than valuation finance metrics (Corrado & Hulten, 2013; Horne, 1995; Mauboussin & Callahan, 2020).

Third, intangibles are bound to have *spillover*: it is easier for other entities to create copies of intangibles than of tangibles; thus, spillover represents “incomplete excludability” (Romer, 1990, p. S75).

Fourth, intangibles can benefit more from *synergies* with one another than tangibles, e.g., by combining multiple reusable software assets in one project. Making use of this fact represents a strong competitive tactic that expedites innovation. It can create powerful incentives both for the companies involved and individual experts. Therefore, intangibles stimulate a fast-moving business that is less based on owning intangibles but on quickly combining them or their spillover to benefit from strategic synergies.

While the global trade in intangibles has yet to be fully understood and reflected in international trade guidelines (Fu & Ghauri, 2021), it can change capitalism (Haskel & Westlake, 2018) unpredictably. Retaining software ownership forever gives the IP owner control over global value chain activities (Fu & Ghauri, 2021). A few companies could gain monopolistic intellectual power and become technological giants (Rikap & Lundvall, 2020), benefitting from licence fees as a constant source of income.

2.2.3 *Assetisation*

The preceding section delved into the multifaceted realm of intangible assets, specifically focusing on their pivotal role in enhancing organisational performance. Intangible assets, notably computer software, possess unique characteristics that have been the subject of research, as outlined in the literature. Their impact on innovation, efficiency, and

competitiveness has been recently acknowledged, highlighting the significance of understanding their value and management within contemporary organisations.

This section focuses on a critical concept that intertwines with the dynamics of intangible assets - 'Assetisation.' It will examine how transforming knowledge and other resources into revenue-generating and tradable assets can revolutionise the landscape of technoscientific capitalism. Assetisation extends beyond the conventional notions of commodities and markets, offering a fresh perspective on how unique, monopolistic control over assets can lead to economic rent extraction and enduring value. To comprehensively grasp the intricacies of assetisation, key aspects that set assets apart from other entities must be considered, such as ownership, control, rentiership, return on investment, and asset value. Moreover, this section will explore the dynamic nature of asset valuation and how it can be influenced by various factors, shedding light on the nuanced strategies organisations employ to maximise the value of their assets. The aim is to provide a holistic understanding of assetisation and its implications for the evolving landscape of knowledge-based economies.

According to Birch (2017, p. 469), *assetisation* is “the transformation of something e.g., knowledge, into a revenue-generating and tradable resource” called an asset. There is no inherent quality in a thing that defines it per se as an asset. For example, infrastructure, nature, publics or knowledge can be turned into an asset (Birch & Muniesa, 2020). Birch (2017) is focussed on revenue and does not comment on savings or other benefits assetisation may bring.

Assetisation is important because “the dominant form that technoscientific capitalism affords is not the commodity but the asset ... not the market speculation but the capital investment” (Birch & Muniesa, 2020, p. 10). Marginal production costs close to zero terminate any further profit increase. However, “assets and assetisation are an empirical ‘blind spot’ in the sense that they are not afforded the systematic and sustained analytical attention they deserve in the contemporary period” (Langley, 2020, p. 1).

Birch & Muniesa (2020) highlight seven aspects that set assets apart from other entities: ownership, control, rentiership, capitalisation, demand, return on investment and asset value.

“Assets are legal constructs, in that *ownership* and control rest on the state enforcement of property and control rights” (Birch & Muniesa, 2020, p. 14). This is in line with what Haskel & Westlake (2018) say. Atasu (2021) uses the term access instead of control. Securing ownership rights can be a lengthy and costly process. Economic value arises from using the asset again and again. Thus, missing intellectual property rights can hinder the reuse software assets (Whalen et al., 2018).

Assets “often involve forms of ‘*rentiership*’ in which monopoly control ... enables the extraction of economic rent” (Birch & Muniesa, 2020, p. 14). Assets can be *capitalised* as a revenue stream, often involving the valuation of discounted future earnings in the present. This differentiates reusable software assets from software products whose present price is based on many small earnings whereas reusable software assets generate few but bigger revenue events which can be distributed over a long time.

“As a result of being unique or constructed monopolies, assets have distinct supply and *demand* logics in which rising asset prices do not lead to new producers or creators entering the market and thereby lowering prices” (Birch & Muniesa, 2020, p. 14). Assets differ from commodities in terms of their demand logics (Birch, 2017; Birch & Tyfield, 2013). As demand rises, prices for commodities tend to decrease. In contrast, the prices for an asset seem to rise as demand increases. Birch and Tyfield (2013) argue that this is due to the assets being unique and hard to replicate. Considering assets as unique is in line with (Haskel & Westlake, 2018). However, Haskel & Westlake (2018) foresee the possibility of replicates due to spillover effects. So, the statement that asset prices rise as demand increases is limited only to those cases when assets represent monopolies, e.g., the copyright of an individual pop song. This monopoly status only applies to software assets that cannot be easily replicated.

The “asset value can be discounted in light of forward-looking expectations about future *returns on investment*, whether or not those expectations are met” (Birch & Muniesa, 2020, p. 14). The discounting is based on future expected uncertainties. While assets can be bought and sold, the emphasis is on the durable economic rent resulting from owning and controlling the asset and limiting access to it (Birch & Muniesa, 2020).

Finally, the *asset value and valuation are dynamic*. “Asset prices and valuations are subject to the actions of owners who may seek to reduce the economic value of their assets, or turn an asset into another form, transfer ownership, or use it to attract new

partners” (Birch & Muniesa, 2020, p. 15). This means the price of a reusable software asset should not be mathematically derived and frozen. Instead, it is situation dependent which links with the dynamic capability theory recognising an organisation’s capability to adapt (Marrucci et al., 2022). Jarvis (2009) describes how certain IT players like Google or Facebook offer some of their IT services for free.

Assets, including knowledge assets, are a development methodology construct, a business strategy – and a legal construct. It is key “*how assets are governed and managed within organisational entities*, rather than how [the asset value] is constituted or represented by biological matter, liveliness, surplus life and so on” (Birch, 2017, p. 470). Concluding based on Birch’s fundamental findings on intangibles, clever software asset management, e.g., in the form of systematic software asset reuse, can increase the value of a software asset.

2.2.4 Servitisation

The preceding section explored the concept of assetisation, a fundamental process in contemporary technoscientific capitalism. Assetisation involves the transformation of various entities, including knowledge, into revenue-generating and tradable resources known as assets. As the discussion of intellectual capital and related concepts delves deeper, it is crucial to recognise that assetisation is pivotal in shaping modern business dynamics.

Assetisation not only underscores the importance of turning intangible elements into valuable assets but also highlights the unique characteristics that set assets apart from traditional commodities. These characteristics, including ownership, control, rentiership, capitalisation, demand dynamics, return on investment, and asset valuation, form the core of our understanding of how intangible resources can be harnessed for economic gains.

Diving in this section deeper into the intricacies of servitisation, will explore how this transformative concept amplifies the impact of assetisation, emphasising the role of services in extending the lifespan and utility of assets. Additionally, this section will delve into the synergies between assetisation and servitisation, highlighting their potential to create novel business models and generate additional value for all stakeholders involved.

Exploring the natural progression in contemporary business strategies crosses the concept of servitisation. Assetisation and servitisation are two complementary concepts. While

assetisation refers to the process of transforming an item into a financially relevant object, servitisation shifts the trend from selling products to selling services. This paradigm shift has profound implications not only for business models but also for the dynamics of value creation and the management of intangible assets. This is because servitisation “turns the incentive for product durability and upgrading upside-down, shifting companies’ focus from volume (i.e., selling ‘widgets’) to performance (i.e., selling the function of that widget)” (Lacy et al., 2020, p. 25). This is also true for reusable software assets. It has wide ranging effects, e.g., on the pricing model, the ownership of operational data but also on the length of use (Lacy et al., 2020).

In essence, while assetisation empowers organisations to recognise the hidden value within their intangible resources, servitisation redefines how that value is delivered and monetised. Together, these concepts present a holistic perspective on the modern economy, where the interplay between tangible and intangible assets shapes new possibilities for innovation, competitive advantage, and sustainability.

Services resulting from tangibles can be turned into or constructed as intangible assets. The logical connection between servitisation and assetisation is that they can be used jointly to create new business models for intangibles thus generating additional value for all stakeholders involved. Assetisation requires servitisation – *providing a flexible service to the client* instead of handing over a suitable commodity to be serviced by the client on his own. This means, instead of selling software, software can be provided as a service, for example, Software as a Service (SaaS) that can run on the cloud (Ismail, 2019).

Servitisation typically intensifies the use of an item because the sharing aspect ensures the use of an item during times it would normally sit idle if it had only one specific user (Lacy et al., 2020). This means reuse *extends the length of use*. The influence of time makes an asset a bet on the future. While awaiting future service revenue, the service provider has to “internalise all the cost for risk and waste” over the lifetime of the reusable item (Press room European Commission, 2014, p. 2).

2.3 Circular Economy

Section 2.3 explains the concept of reuse and contrasts it with consumption-based organisational performance.

2.3.1 Reuse

Our planet comprises a finite number of resources. The circular economy allows a closed economic system centring on circular flows of resources (Del Vecchio, 2022). It is based on actions maintaining value, quality and quantity of stock. These comprise circular economy concepts like reusing, repairing, remarketing or remanufacturing (Stahel & MacArthur, 2019), refurbishing or recycling - see Figure 2.7. They can be applied along an asset lifecycle of input, production, distribution, use, disposal, and waste and emission handling (Geissdoerfer, 2020; Press room European Commission, 2014).

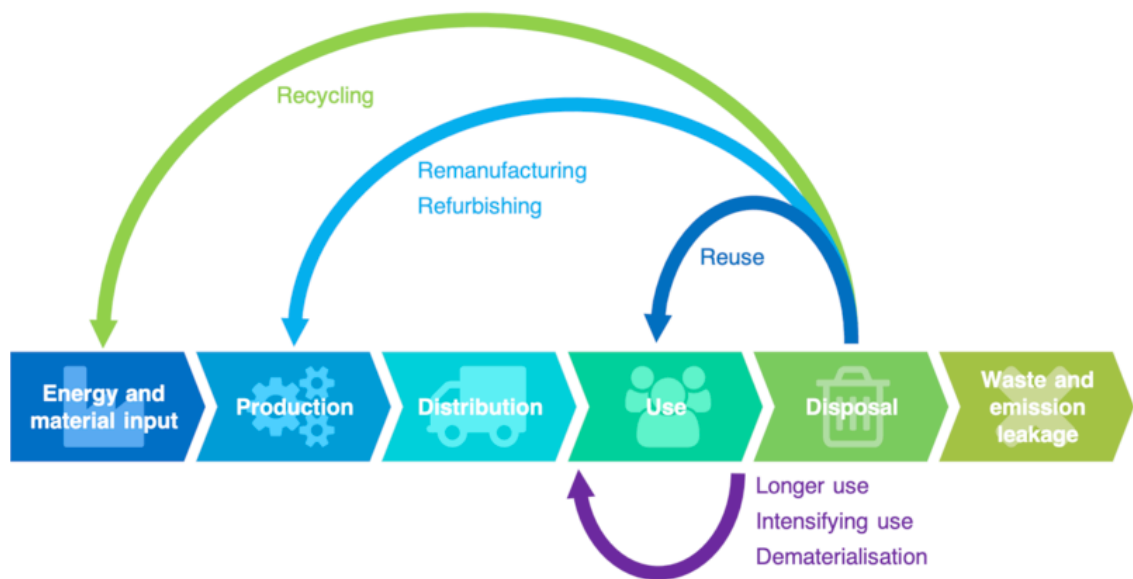


Figure 2.7 Reuse as part of the circular economy (Geissdoerfer, 2020; Press room European Commission, 2014)

Current circular economy literature is predominantly focussing on tangible resources. While all circular economy concepts are relevant in sustainable resource management, the concept of reuse is particularly important for software assets due to their intangible characteristics.

Software assets do not wear out over time the way tangibles do. Software can be copied and used infinitely without a substantial loss in quality. The effect of wear or tear does not apply. From this point of view, circular economy concepts such as reducing, repairing, recycling or remanufacturing are less relevant in the context of software assets.

The software reuse literature is inconsistent and misses details in the definition of reuse. The term reuse is often applied in the sense of using again “an object without any

reprocessing or treatment” (Sinha & Modak, 2021, p. 1411). Although seemingly straightforward, the terms use versus reuse hold distinct connotations and implications when applied to various domains, particularly in the context of assets, knowledge, or resources. In the context of this thesis, it is essential to establish a clear understanding of the two fundamental concepts.

The term *use* refers to the initial, primary consumption or application of an asset or resource for its original purpose, fulfilling an intended function. This utilisation occurs without alteration or multiplication.

Reuse involves repeatedly applying a resource, using an asset “more than once” (Zabardast et al., 2023, p. 1), to gain additional value. Existing circular economy literature focuses predominantly on tangible reuse, where the user is the owner who takes action to expand the asset's lifetime – leaving a gap for all other possible combinations. Data reuse is implicitly defined as using data collected from another person (Tenopir et al., 2020). The characteristics of the utilisation situation determine whether it’s a use or a reuse.

Literature misses explaining that reuse does not aim to extend the lifetime of a broken software asset. Instead, reuse can be applied to create one or more parallel instances of intact intangibles to potentially serve many business clients simultaneously to increase efficiency or revenue. E.g., the software asset is in use at Schiphol airport and at the same time at Heathrow Terminal 5 and DHL in Germany. It can be argued that running the software in three situations in parallel expands the asset's lifetime overall.

Schloss (2018) researching microbiology and Cacioppo (2015) researching social, behavioural and economic sciences explain the important underlying concept of reuse as shown in Table 2.1.

Table 2.1 Grid-based system for defining reproducibility, replicability and generalisability/reuse - adapted from (Cacioppo et al., 2015; Schloss, 2018)

Item functions	Same experimental system	Different experimental system
Same item functions	Reproducibility	Replicability
Different item functions / variants	Robustness	Generalisability/Reuse

According to them, reuse happens in a different experimental system. It requires different item functions or variants (see bottom row of Table 2.1) (Krüger & Berger, 2020). This is a fundamental literature finding with far-reaching implications. If each implementation of the reusable software asset typically has different, individualised feature functions, it can serve market *niches*. According to Jarvis (2009, p. 27), “Serving masses ... is no longer the be all and end all of business. Serving targeted masses of niches ... is the future.” This niche effect has nevertheless not been discussed in the software asset reuse literature yet.

Reproducibility, replicability and robustness also listed in Table 2.1 are not in the focus of this research.

No literature could be found on how these different experimental systems, the reuse situations, are interrelated, or how they could be classified or described logically.

Reuse requires a conscious decision. There is always a moral aspect to taking a reuse decision and the decision to act depends on the expected value (Gouinlock, 1995).

The underlying reason for reuse is scarcity (Salgot et al., 2017; Stahel & MacArthur, 2019) – something essential is available only in a limited quantity (Montani, 1987) at a certain time or place. Although knowledge is infinite (Percival, 1996) there can be a temporary or local shortage. This thesis argues, reuse allows a precious resource like knowledge to be better managed when it is temporarily or locally limited.

The circular economy shifts to a concept “in which products and materials are recycled, repaired and reused rather than thrown away, and in which waste from one process becomes an input into other processes” (Preston et al., 2019, p. 2). It *contrasts with our current take-make-dispose business* principle based on throughput maximisation and low labour prices (Fivet & Brütting, 2020). Moving from a market-based view towards a *resource-based view* (Atasu & Van Wassehove, 2021) focusing on value preservation (Fivet & Brütting, 2020) requires the right circular business model. Atasu et al. (2021) recommend basing the circular business model on three fundamental strategies for circularity: retain product ownership via rents or leases, product life extension (Mihale-Wilson et al., 2021; Sinha & Modak, 2021), by designing products to last, and design for circularity. These three principles apply to reusable software assets, too. Table 2.2

highlights the resource-based view as reuse is dependent on the availability of valuable assets and other unique resources that provide a competitive advantage.

The circular economy can be characterised by consumption simplicity, accessibility, waste minimisation and green excellence during value creation (Sidmou, 2021). On the other hand, Spilhaus (1971, p. 326; italics in the original) wrote from a consumer’s perspective, “one consequence of recycling and reuse is that we will own nothing except works of *art*. We must get used to the idea that we are no longer consumers and there is no longer ownership. We must replace *usership* for ownership”. Replacing the traditional capitalist concept of ownership with a concept of usership or control which is not yet fully explored, may cause political and economic disruption. Assetisation and servitisation clearly require a different approach compared to the sale of commodities. Table 2.2 summarises this discussion by contrasting reuse with consumption.

Table 2.2 Summary of consumption versus reuse (source: author)

Consumption	Reuse
Value creation	Value preservation
Market-based view	Resource-based view
Throughput maximisation	Waste minimisation
Take-make-dispose business based on low labour prices	Expanding the lifetime of a component
Low cost, high profit	Design to last/design for circularity
Linear economy	Circular economy
Commodity	Reusable software asset
Ownership (which allows usership)	Usership (by paying rents/leases to an owner)
Process innovation	Product innovation

As discussed in the example of the luggage handling software in Section 1.3, adapting a reusable software asset increases its value and expands its usage to new purposes so that new use cases can be targeted. This requires creativity (Wegener, 2016), innovation (Stahel, 2016) and abstraction (Biggerstaff, 1994) to enable repurposing or knowledge brokering (Wegener, 2016).

In this context, “the creative act is the process of moving ideas from where they are known (and maybe categorized as useless or trash) to where they are not” (Wegener, 2016, p. 6). While it is sometimes possible for reuse to occur locally among people in very similar situations, the real benefit of software asset reuse unfolds if a bigger set of clients can be reached. “Revolutionary innovations often come from very evolutionary origins” (Hargadon, 2002, p. 51).

Reuse requires customer acceptance (Whalen et al., 2018). This acceptance goes beyond the concept that what is received is “not new”. Reuse involves prerequisites, e.g., that the ownership and control of the intangible asset remains with the IT provider. This requires care, in the sense of maintenance, and trust (Stahel & MacArthur, 2019).

2.3.2 *Economy of Reuse*

For most reuse projects, economic factors are paramount for success (Salgot et al., 2017). In a financially competitive market, full-cost recovery is often required, i.e., the targeted end user has to pay – despite the fact that “the benefits derived from reuse are not only for the end users but also for all the users” in the future (Salgot et al., 2017, p. 269). This means IT providers often expect asset teams to generate revenue and savings from the first reuser in a magnitude that pays off all costs. That is because software asset reuse involves an element of future uncertainty – it is unclear how many more reuses will be happening in the future.

Reuse generates non-monetised benefits such as reliability, reputation, self-sufficiency in terms of safety of supply, environmental effects and improved quality. At the same time and in contrast, reuse causes non-monetised costs such as an effect on the overall carbon footprint, perception of reduced quality, internal effects downstream such as increased complexity of the system, quality impacts when errors are carried over, threat of supply and social aspects (Salgot et al., 2017). These *non-monetised benefits* and costs play a role and cannot easily be dealt with in financially driven businesses.

While mass production can occur, mass reuse is difficult to achieve in practice. Reuse implies a high degree of manual intervention “to adapt ... to a new application context, and this can be subtle even in the simplest of cases” (Biggerstaff, 1994, p. 6). This can be a showstopper in reuse (Whalen et al., 2018). In addition, it can be challenging to pull together experts in an ad hoc way when a new experimental reuse situation arises. While standardisation may help (Copello et al., 2021), reuse *typically lacks efficiency*.

“B2B sharing might present the most promising opportunities for large corporations, especially for businesses which maintain high-cost assets that have low utilization rates” (Lacy et al., 2020, p. 24). This is because reuse provides access to valuable intangibles at a very attractive price. Additionally, reuse helps to maintain a valuable and scarce resource under control (Fivet & Brütting, 2020; Salgot et al., 2017; Stahel & MacArthur,

2019). The fact that software asset reuse targets B2B clients is a new finding that has not been mentioned in the software asset literature to date. Later sections of this thesis detail this point.

Several reuse management theories apply, such as earlier mentioned Barney's (2001) resource-based view theory and Stahel's (2019) circular economy theory. Further, Rogers' Diffusion of Innovation Theory which focuses on how innovations are adopted and diffused within a social system along its limitations (Talwar et al., 2020), Granovetter's and Burt's (Vedres, 2022) Social Network theory on how social networks and relationships influence behaviour along with further other theories come into play.

2.4 Knowledge Management

For a long time, researchers have discussed knowledge management and intellectual capital as two separate entities (Paoloni et al., 2020). The literature on intellectual capital covers the value of intangible assets, whereas knowledge management focuses on management practice. Only recently have scholars denoted intellectual capital as all the knowledge that business organisations can use to gain financial benefit (Hussinki et al., 2017; Youndt et al., 2004).

Software assets represent knowledge in the form of applied software process knowledge and business process/domain knowledge (Di Cosmo & Zacchiroli, 2017; Dreyer & Wynn, 2016; Huang, 2019; Kneuper, 2002). Therefore, it is useful to first understand the general management of knowledge before looking into the specifics of managing a reusable software asset.

Section 2.4.1 discusses various types of knowledge. Section 2.4.2 compares knowledge against information and data. Section 2.4.3 covers the knowledge work context and contrasts explicit versus tacit knowledge. Subchapter 2.4 concludes by detailing the four key knowledge processes: knowledge creation, knowledge storage, knowledge distribution and knowledge application, including reuse.

2.4.1 Types of Knowledge

“Knowledge is defined variously as (i) expertise and skills acquired by a person through experience or education; the theoretical or practical understanding of a subject, (ii) what is known in a particular field or in total; facts and information or (iii) awareness or

familiarity gained by experience of a fact or situation” (Matallo, 2021, p. 2). This is one of the most recent definitions of knowledge.

The oldest available definition of knowledge comes from the Greek philosopher Plato (427–347 BC). He said, “knowledge is more honourable and excellent than true opinion, because fastened by a chain” (Conford, 1935, p. 4). This implies that evidence raises knowledge above opinion, leading to the term “Justified True Belief”, JTB, (Nozick, 1981, p. 267). History has shown that even science is changing over time so that no knowledge can be seen as fully justified. “The production of each empirical finding should be viewed more as a promissory note than a final conclusion.” (Cacioppo et al., 2015, p. 2).

Parikh and Renero (2017, p. 100) see knowledge as “agent relative” – it is subjective, person-specific. Wijnhoven (2008) lists subjectivism and objectivism as two major knowledge paradigms. Adopting a pragmatic point of view of here and now, knowledge can only be seen as relatively true.

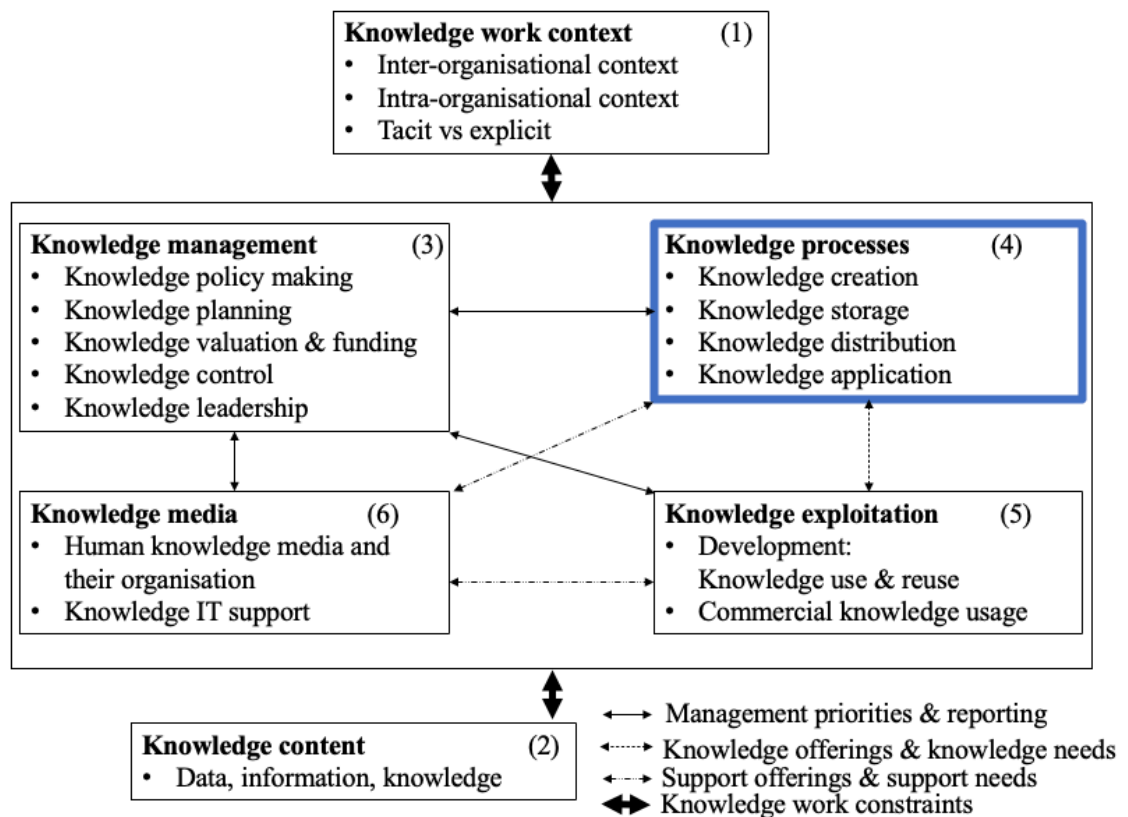


Figure 2.8 A model of knowledge work - adapted from (Wijnhoven, 2008)

Pragmatism works reasonably well as long as “Knowledge is based in truth.” (Parikh & Renero, 2017, p. 5). The limiting factor is that differentiating between true and false evidence requires further knowledge that may not be available. Therefore, Spender (2008, p. 163) suggests the need to “consider the different types of knowledge available ... and to compare and contrast or even relate them”. While scientists have classified knowledge in a variety of ways, Wijnhoven (2008) took a particularly structured approach by mapping the types of knowledge against (1) knowledge work context and (2) knowledge content and by identifying four major domains of knowledge work: (3) knowledge management, (4) knowledge processes, (5) knowledge exploitation and (6) knowledge media (see Figure 2.8).

For the *knowledge work context*, there is an intra-organisational, hierarchy-dictated aspect to consider. Inter-organisational knowledge – between two or more organisations – is governed by market principles, network and hierarchy – comprising, particularly, explicit and tacit knowledge that plays a crucial part in this research and is further detailed in Section 2.4.3.

As for *knowledge content*, it can be distinguished as data, information and knowledge (see Section 2.4.2).

When it comes to *knowledge management*, Wijnhoven (2008) differentiates between knowledge management in the intra-organisational and in the inter-organisational context. Strategic and tactical knowledge activities fall into the category of intra-organisational knowledge management. Examples of strategic knowledge activities are knowledge acquisition and collaboration policies, IP rights and exploitation plans. IP rights are relevant for this work when asset reuse results in the generation of IP licences. Digital literacy strongly impacts knowledge management (Jasin et al., 2024).

Knowledge management in the inter-organisational context deals with operational *knowledge management processes* related to exploiting knowledge products and reusable software assets. Operational knowledge management processes form the centre of this research and are detailed in Sections 2.4.5–2.4.8.

Wijnhoven (2008) lists *knowledge exploitation* as a separate knowledge work context because it explains why all knowledge work is being done – it allows income to be generated (as detailed in Figure 2.4 in Section 2.2.2).

There are two *main knowledge media* – human and IT, which are discussed in Section 2.4.6.

2.4.2 Knowledge Content

Figure 2.9 depicts the relationship between data, information and knowledge. *Data*, “a set of discrete, objective facts about events” (Davenport & Prusak, 2000, p. 2), can be transformed into information. According to Fadler and Legner (2020), based on Zechmann’s work (2016), data are an intangible asset if they are *identifiable, non-monetary* and *without physical existence*, have the *potential to generate economic benefit* for the respective business organisation, have procurement or production *cost* that can be quantified and their *future benefit* can be gathered and restricted in access.

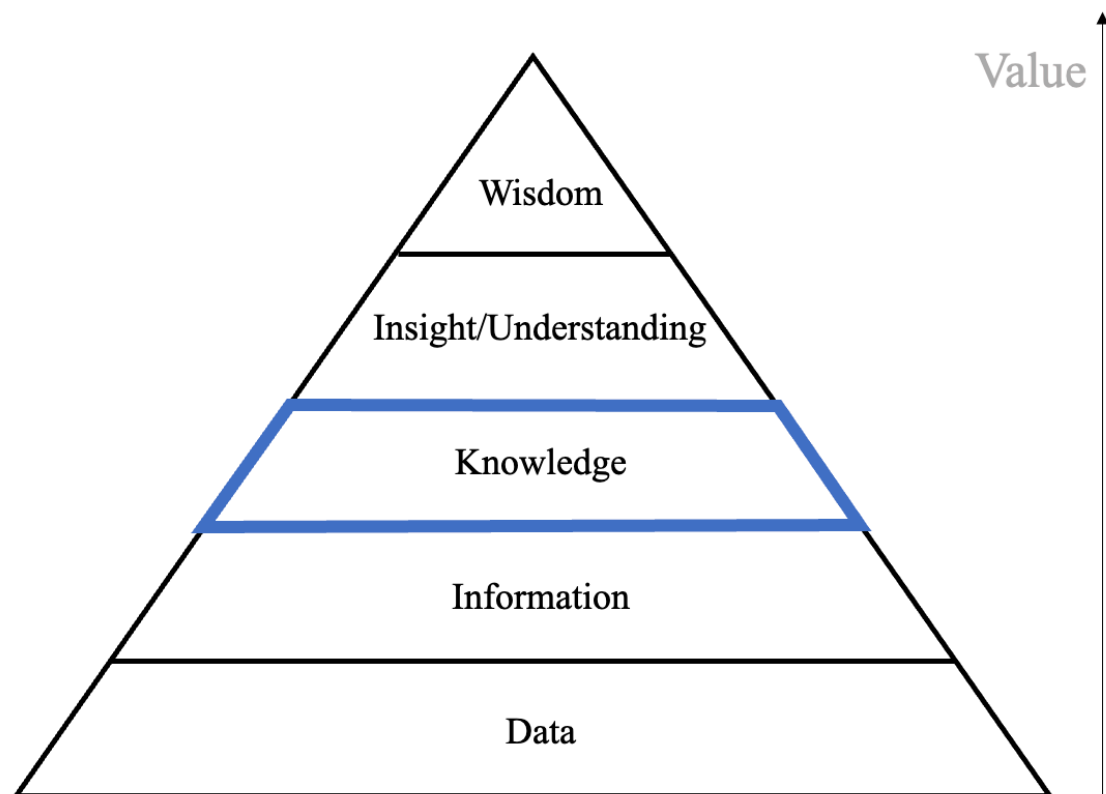


Figure 2.9 Relationship between data, information and knowledge based on Russell Ackoff’s Apex - adapted from (Krishnaveni & Raja, 2012; Oppenheim et al., 2003)

Information represents data packaged in a context (Bouthillier & Shearer, 2002). Oppenheim (1998, p. 212) was one of the first to define information as an asset by saying it is “the stock of information stored plus accumulated expertise in accessing and interpreting information”. This definition involves three crucial aspects: having *information*, being able to *localise it* and *interpret/use it*. This localisation is the main

reason why undocumented knowledge cannot be an asset. According to Abeysekera (2021), information is knowing-of, whereas knowledge is know-how.

Information can be further transformed into knowledge while its value to the business increases steadily (Oppenheim et al., 2003). The clear differentiation between data D, information I, knowledge K and wisdom W, e.g. in the form of the DIKW hierarchy (Rowley, 2007), goes beyond philosophical considerations.

Knowledge provides instruction-like answers to how-to questions (Ackoff, 1989). Software instructions contain explicit knowledge giving a machine-readable answer to a how-to question. Haskel and Westlake (2018, p. 64) define “*knowledge* as connections made between pieces of information, supported by evidence, to form a coherent understanding. Knowledge cannot exist without information and knowledge is required to fully understand and interpret information”. Ackoff’s pyramid places *understanding*, which provides explanations and answers why questions, above knowledge (Krishnaveni & Raja, 2012). *Wisdom* (Bangen et al., 2013) forms the top of the knowledge pyramid. It addresses effectiveness and represents values, including the reasoning from emotions, which cannot be machine-processed (Ackoff, 1989). Figure 2.10 graphically summarises how data, information, knowledge, insight and wisdom differ from each other.

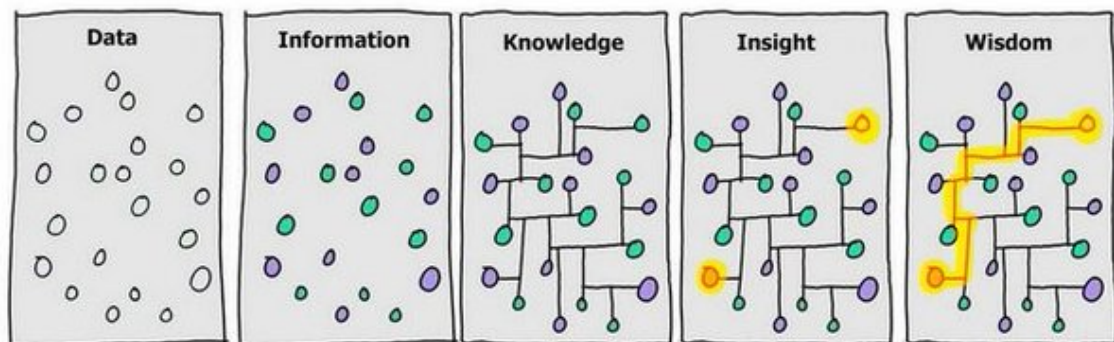


Figure 2.10 Difference between data, information, knowledge, insight and wisdom (MacLeod & Sommerville, 2016)

“Knowledge is something intrinsic in the human mind which cannot be directly communicated” (Oppenheim et al., 2003, p. 160). While data D and information I can be passed from one person to another, knowledge K is the result of individual information processing within each single person (see Figure 2.11).

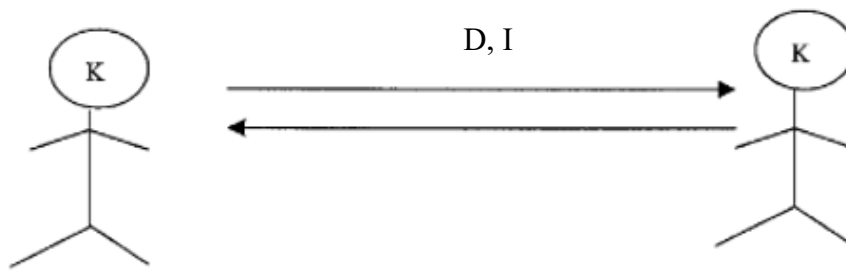


Figure 2.11 Knowledge transfer (Oppenheim et al., 2003)

2.4.3 Knowledge Work Context

Coded knowledge e.g., software, drawings, text, is classified as an *explicit knowledge* (Gamble, 2020; Majchrzak et al., 2004). Explicit knowledge represents formally documented public information. It is acquired by a cognitive learning process, corresponds to contextualised information and forms only a small proportion of human knowledge (Wang & Yang, 2015). Explicit knowledge can be detached from the knowledge owner and thus *shared with another person* (Burciu & Kicsi, 2015; Nonaka & Teece, 2001). This is a pre-condition for reuse. In contrast, *implicit knowledge* refers to applied information that has the potential to be articulated but has not been made explicit (Nakamori, 2020; Nickols, 2000).

Tacit knowledge differentiates a novice from an expert. It covers personal experience like insight and is typically shared face to face (Nakamori, 2021). It is non-rational and intuitive (Burciu & Kicsi, 2015) and can hardly be documented (Nickols, 2000). Therefore, it can neither be distributed nor reused.

Just as individuals can gain knowledge, organisations, i.e., groups of people, can gain knowledge as well. As shown in Figure 2.12, socialisation, externalisation, combination and internalisation are organisational learning patterns that allow the creation of knowledge and the transformation of knowledge from tacit into explicit and vice versa (Dreyer & Wynn, 2016; Krishnaveni & Raja, 2012; Nonaka, 1991; Nonaka & Teece, 2001).

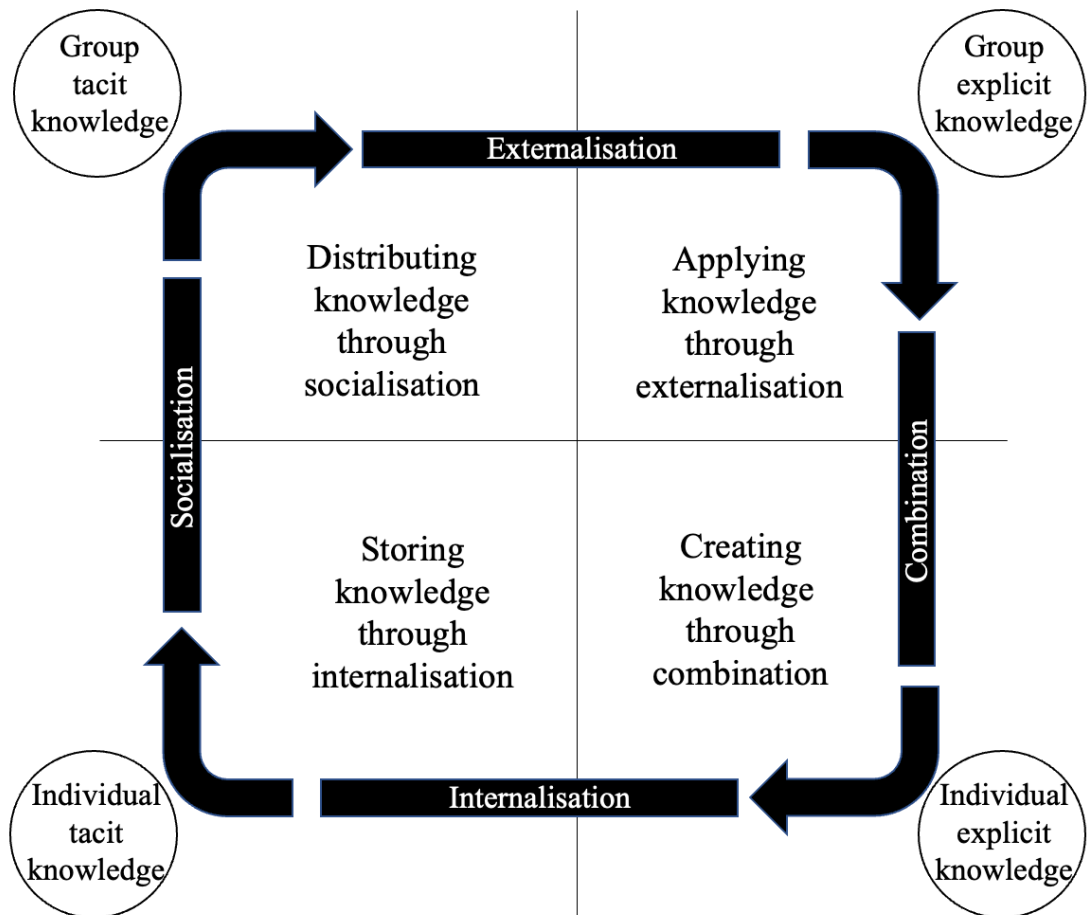


Figure 2.12 Socialisation, externalisation, combination and internalisation (SECI) - adapted from (Dreyer & Wynn, 2016; Nakamori, 2020; Nonaka & Teece, 2001)

Externalisation, transforming individual subject matter experts' tacit into explicit knowledge, is a fundamental step in every software development project (Dreyer & Wynn, 2016). It is significant for this research since only explicit knowledge can be machine-processed (Fakhar Manesh et al., 2021; Haskel & Westlake, 2018) and reused.

Team-internal transactive processes lead to logically organised knowledge in the mind of the team members (Zhou, 2019). This encourages specialisation in the team and requires coordination in terms of who knows what. Team-internal knowledge build-up via collective thinking (Borge et al., 2018; Schultz, 2022) increases credibility and trust among team members (Human, 2020). According to Zhou & Pazos (2020) the build-up of a team's transactive memory system (TMS) increases the team's affective outcomes – trust, efficacy and satisfaction – as well as the team's behavioural outcomes – learning, knowledge sharing and creativity. This is required to reuse software assets in global IT organisations. Given the acquired knowledge, the asset team as such can be seen as a valuable asset, a centre of competence (Ram & Devi, 2019).

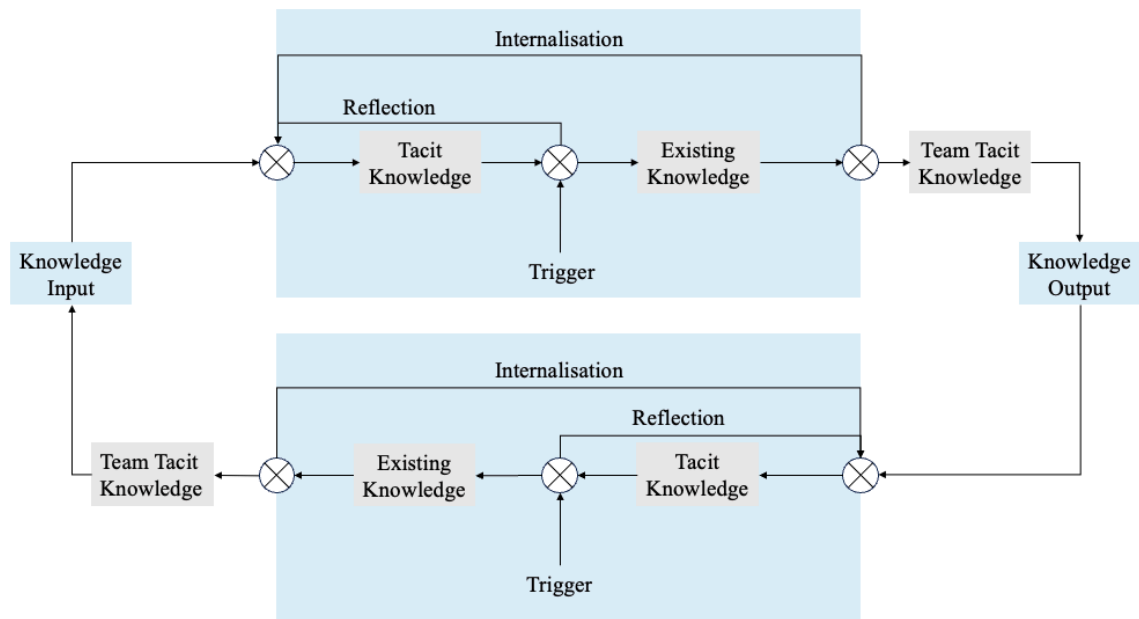


Figure 2.13 Individual and group tacit knowledge process – triggers are represented by a crossed-out circle (Dreyer & Wynn, 2016)

Figure 2.13 depicts the knowledge process, putting emphasis on triggers. The top of the figure shows the knowledge process of individuals and, at the bottom, those of teams. These processes do not run in an autarchic way but require a stimulus, a trigger – a concept covered in Section 2.4.8.

2.4.4 Management of Knowledge Processes

Peter Drucker was the first to describe the knowledge worker (Drucker, 1957). His work led to the emergence of the knowledge management (Nakamori, 2020), thus shifting away from the pure organisation management (Spender, 2008).

Different researchers have identified different knowledge management processes – see the header row of Table 5.1 in “Appendix A: Operational Knowledge Processes”. In the first column, the author of this research chronologically lists 25+ research papers plotted against 20+ research-covered, logically horizontally arranged knowledge management sub-processes. Following Wijnhoven (2008), Alavi & Leidner (2001) and others, the researcher clustered these sub-processes into four fundamental operational knowledge management processes (see bottom row of Table 5.1) that summarise how knowledge is processed: *Knowledge creation* (Section 2.4.5), *knowledge storage and retrieval* (Section 2.4.6), *knowledge distribution* (Section 2.4.7) and *knowledge application*, including reuse (Section 2.4.8). These are vital for this research and will be looked at in detail.

2.4.5 Knowledge Creation

Knowledge creation (Song et al., 2007) comprises what other authors describe as the acquisition (Fakhar Manesh et al., 2021), identification (Alavi & Leidner, 2001), discovery (Schwartz, 2005) or capturing (Prusak & Davenport, 1998) of knowledge. Knowledge creation is relevant in IT businesses and quantifies and conceptualises knowledge for assisted, automated, automatic or autonomous flow (Succar & Poirier, 2020).

The act of knowledge creation is closely linked with the “processes of imagining, learning, and forgetting” (Spender, 2008, p. 165). Forgetting is the – mostly unconscious – sorting out of knowledge. Learning is gluing the pieces of knowledge together to allow knowing. While learning can be done individually, it is typically a social process (Auerbach, 2016). Knowledge creation has been detailed by Nonaka (1994) and subdivided into socialisation, externalisation, internalisation and combination (see Figure 2.12).

Knowledge is created in a specific environment under set conditions at a certain time. Two back-and-forth knowledge-creation mechanisms are of relevance for this research. As illustrated in Figure 2.12, there is a constant interchange between group knowledge and individual knowledge. Group knowledge helps in the creation of individual knowledge and vice versa.

According to Nakamori (2020, p. 63), “the source of organisational knowledge creation is the mutual complementation and circulation of tacit knowledge and explicit knowledge”. The more the knowledge circulates between externalised and internalised, the richer the resulting knowledge. So, alternately describing and doing (see Figure 2.14) broadens the knowledge and enables innovation (Cook & Brown, 1999; Gourlay, 2006; Nickols, 2000).

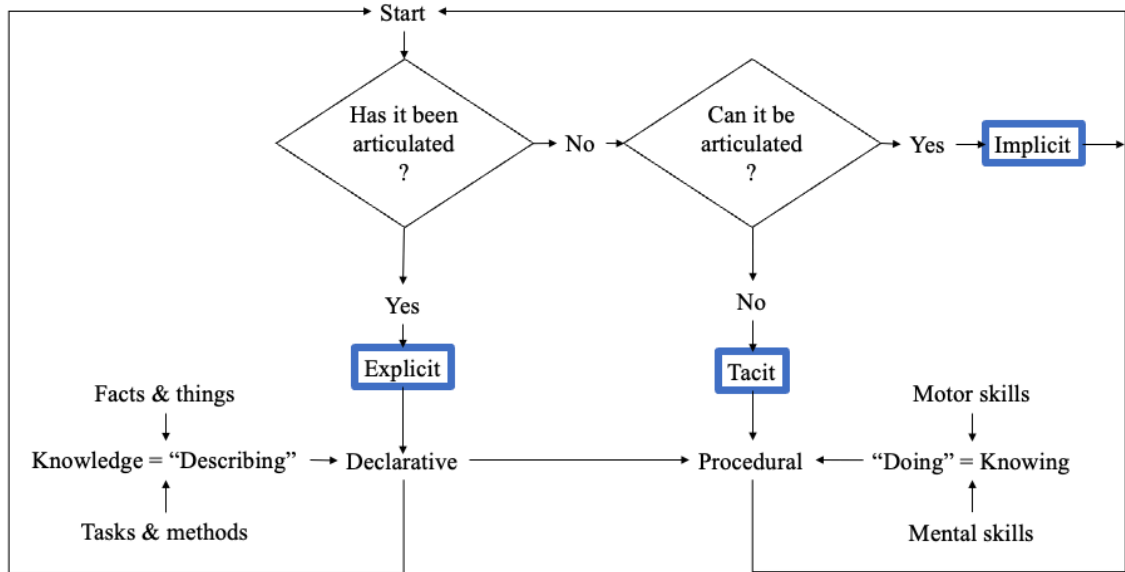


Figure 2.14 Relationship between explicit, tacit and implicit knowledge - adapted from (Cook & Brown, 1999; Nickols, 2000)

Researchers have investigated the factors that influence knowledge creation. Following Table 2.3, factors and related processes like competency & skills of the individual expert, a shared leadership vision as well as an atmosphere of trust and collaboration are most helpful in creating knowledge. Both Ueki (2011) and Koloniari (2019) describe a strong leadership relating to the transactional leadership theory (Alrowwad & Abualoush, 2020) as of which leaders set clear objectives and punish or reward teams based on meeting set expectations.

What is rarely covered in the literature is a focus on the *trigger mechanism* that initiates the systematic creation of knowledge as depicted in Figure 2.13. Likewise, not mentioned in the literature is the process of *knowledge retrieval for level setting*, which should be the initial step before any knowledge creation as it is fundamental for later knowledge reuse. This level setting informs if the to-be-created software asset already exists. According to Stenholm et al. (2019), it is typically not established as a business process as it has neither a tangible outcome nor does it gain any immediate benefit. Introducing the step of conscious level setting would require changing the behaviour of the asset creator.

Table 2.3 Overview of factors impacting knowledge creation (source: author)

Factor	Study (Teerajetgul & Charoenggam, 2006)	(Ueki et al., 2011)	(Koloniari et al., 2019)	(Dong et al., 2015)	Total
Individual competency / skills & expertise / systematic HR development	☛	☛		☛	☛
IT support	○		○		○
Incentives	○				○
Vision of leadership / strategy & planning	○	☛	☛		☛
Improve brand-value and customer satisfaction		☛			
Trust		☛	☛		☛
Information sharing / collaboration	☛	☛	☛	☛	☛

☛ impact; ○ negligible impact

Knowledge creation is closely interwoven with knowledge storage and sharing because it makes sense to keep creation and communication relatively synchronous (Nonaka, 1994) so that the time lapse between these is minimal.

2.4.6 Knowledge Storage

The second of the four critical knowledge management processes covers knowledge storage (Song et al., 2007). It comprises subprocesses like recording, capturing (Alavi & Leidner, 2001; Prusak & Davenport, 1998; Schwartz, 2005), organising, configuring (Wijnhoven, 2008), coding (Schwier et al., 2004), structuring, indexing and interlinking (Masa'deh et al., 2019), representing (Hedlund, 1994), maintaining, managing and retrieving knowledge (Koech Sitienei et al., 2015; Masa'deh et al., 2019).

Knowledge storage and retrieval systems “facilitate and enable knowledge transfer” (Taraszewski, 2017, p. vi). Storing explicit knowledge in repositories (Jasimuddin, 2005; Maccanti et al., 2016) has been driven by the expectation that individuals and teams would be able to find, extract, understand and apply the *stored knowledge time and space independently* which increases the reach of the knowledge (Sahibzada et al., 2019).

Only a subset of human knowledge can be stored – the explicit knowledge. Knowledge on organisational activities must be consciously stored in *organisational memory* for later

retrieval to preserve a company's specific knowledge and enable organisational learning (Abdelwhab Ali et al., 2019; Alavi & Leidner, 2001; de Holan & Phillips, 2004; Stein & Zwass, 1995). Organisational knowledge that is not constantly in use is in danger of being forgotten (Taraszewski, 2017). In contrast to individual knowledge, the organisation (Hjørland, 2013) of organisational knowledge requires special attention to enable the group of potential reusers to locate and understand stored knowledge.

Employee performance is strongly dependent on their ability to benefit from knowledge storage and retrieval (Koech Sitienei et al., 2015). Karimpanal & Bouffanais (2019) demonstrate that upfront knowledge storing, in e.g. self-organising knowledge maps, helps agents significantly in accomplishing business tasks more quickly and well.

Continuous maintenance of stored knowledge is an important aspect of knowledge management that has not been detailed in the literature. It is of relevance for e.g., IT knowledge that is subject to continuous change which relates to the dynamic capability theory (Marrucci et al., 2022).

Knowledge management infrastructure covers technical infrastructure and social infrastructure. The *technical storage* of knowledge is determined by using software and hardware tools that aid storage, preservation and retrieval of knowledge. The *social infrastructure* depends on soft factors such as the *organisational culture*, via rules and guidelines, and the *organisational structure* via roles and responsibilities (Abualoush et al., 2018; Masa'deh et al., 2019).

Taraszewski (2017) has conducted the biggest and most recent study on knowledge storage. As shown in Figure 2.15, he identified five critical success factors for knowledge storage systems based on DeLone and McLean's Information System Success Model (2003). The benefit of any storage system depends on the *use* as well as *user satisfaction*. Use and user satisfaction depend on each other as well as on the *knowledge content quality*, the *knowledge management system quality* and *knowledge management system service quality*.

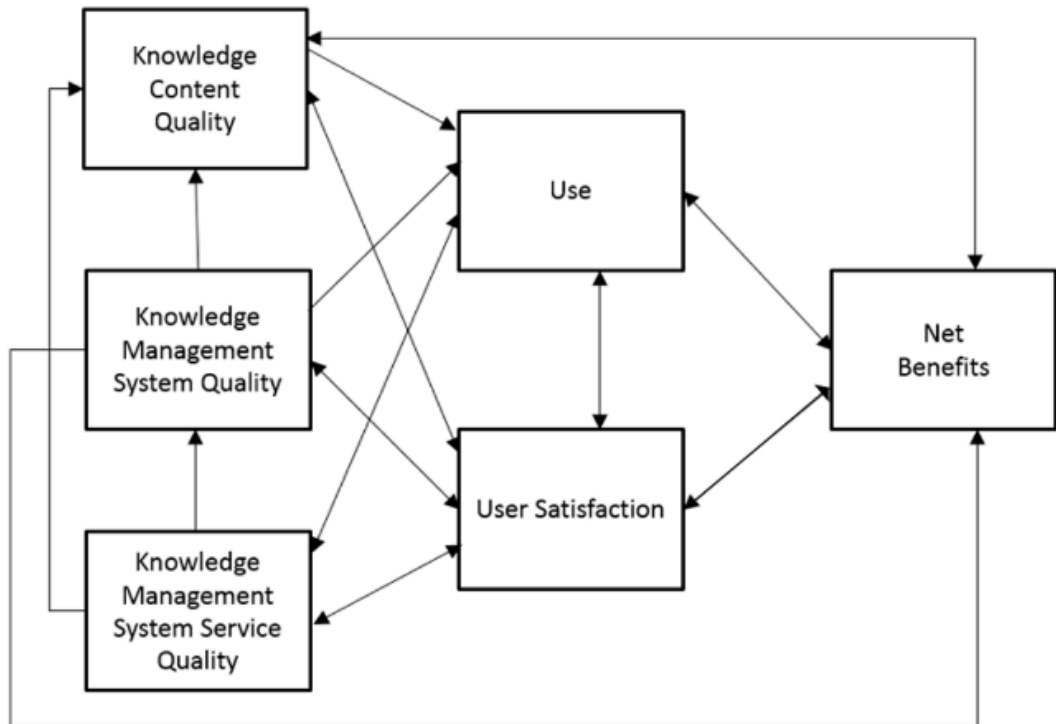


Figure 2.15 Conceptual knowledge storage/retrieval system success model (Taraszewski, 2017)

In addition, Taraszewski (2017) did a detailed analysis and ranked 18 influential storage/retrieval success factors to facilitate and enable knowledge transfer outside the Knowledge Management System. According to this, *Strategy and Leadership* can be seen as fundamental, followed by fostering the willingness of people to *share their knowledge* which is closely linked to the *organisation culture*.

The top three critical success factors of Strategy & Leadership sorted by impact are: the management understands the value of knowledge management, the management’s continuous commitment to resources and top management support (Taraszewski, 2017). All of these factors have an inspiring and empowering character and are therefore aligned with the transformational leadership theory (Alrowwad & Abualoush, 2020).

Further, IT practice experience shows that experts use knowledge stores to locate other experts they contact and talk to – initiating an *interpersonal knowledge transfer* (Terjesen, 2003), thus acquiring unsaved tacit knowledge. These are surprising literature findings given that earlier research tried to increase the reuse of software assets by focusing

predominantly on improving IT technology, which is of lower criticality, according to Taraszewski (2017).

Organising software and related artefacts in a knowledge hub or repository helps to maximise reproducibility and reuse (Ram, 2013). On the downside, enterprises quickly end up with large volumes of information (Nakamori, 2020; Terjesen, 2003) which raises challenges storing, maintaining and retrieving (Reinhartz-Berger, 2024). Next, Section 2.4.7 explains how stored knowledge can be distributed and shared.

2.4.7 Knowledge Distribution

Knowledge distribution (Fakhar Manesh et al., 2021; Hjørland, 2013; Schwartz, 2005; Schwier et al., 2004), the third knowledge management process covers subprocesses like retrieval (Alavi & Leidner, 2001; Fanchao et al., 2006; Schwartz, 2005) and the manipulation of knowledge (Alavi & Leidner, 2001), access to (Song et al., 2007) and sharing (Alavi & Leidner, 2001; Masa'deh et al., 2019; Prusak & Davenport, 1998; Schwartz, 2005; Spender, 2008), dissemination (Eid & Nuhu, 2011), transfer (Wijnhoven, 2008), exchange (Masa'deh et al., 2019), transformation (Hedlund, 1994), modelling (Schwartz, 2005), evaluation (Schwartz, 2005) and interpretation (Huber Jr, 1991; Prusak & Davenport, 1998) of knowledge. The distribution of knowledge fosters the exchange of knowledge, thus generating new knowledge among those involved (Masa'deh et al., 2019). Interpersonal interaction between employees can boost the knowledge distribution (Fayyaz et al., 2020; Koloniari et al., 2019).

The refinement of Shannon's (1948) original *sender-message-channel-receiver* model of communication forms the basis of knowledge distribution. It relies on the value of the input knowledge, the motivation of the sender to share knowledge, the capability of the transmission channel, the motivation of the receiver to listen and the capacity of the receiver (Alavi & Leidner, 2001; Wijnhoven, 2008). While IT can assist in all stages, it takes management attention and dedicated internal company processes to run through all knowledge distribution phases systematically. Irick (2007) explains how communities of practice and a shared workspace can help create and distribute knowledge.

Researchers have identified factors impacting the knowledge-sharing process. Table 2.4 summarises recent relevant studies. Accordingly, knowledge-sharing is mainly driven by mutual trust, the right leadership support, a supportive organisational culture, as well as

social knowledge sharing tools. Fayyaz (2020) states that “upholding a knowledge sharing culture ushers in continuous innovative performance while moving toward open innovation ... knowledge sharing is the lynchpin for organization innovation efficiency”.

Table 2.4 Overview of factors impacting knowledge sharing (source: author)

Study	(Abdelwhab Ali et al., 2019)	(Hejase et al., 2014)	(Akosile & Olatokun, 2020)	Total
Intention	✿			
Trust	○	✿	✿	✿
Motivation	✿			Autonomous job motivation
Leadership support / policy	✿	✿	✿	✿
Rewards	○			
Organisational structure	○			
Organisational culture	✿	✿	✿	✿
Social media / Web 2.0	✿		✿	✿
Knowledge sharing system	✿			

✿ impact; ○ negligible impact

Hejase (2014) highlights in his study the *negative impact of knowledge ownership*. “Scientia potestas est”, which means “knowledge is power”, is attributed to Francis Bacon in 1597 (Rodríguez García, 2001, p. 109) and can cause people to naturally resist sharing their knowledge. This means while knowledge in the form of software is typically used to generate revenue, some software owners could control global business activities and become technological monopolies as discussed in Section 2.2.2.

In contrast to Akosile & Olatokun (2020), Hejase (2014) states transformational leadership beats transactional leadership when it comes to the distribution of tacit knowledge because its success is hard to measure and monitor. The reason is, *intrinsic motivation* gives employees immediate motivation and satisfaction (Hejase et al., 2014).

Motivation matters. Like Hejase (2014), Kuo (2013) identified that controlled job motivation i.e., expecting rewards or avoiding punishment resulting from transactional leadership, has negative side effects such as knowledge hoarding and hiding. In line with this, incentive programmes seem not to generate additional knowledge sharing (Fichman & Kemerer, 2001) but rather most knowledge distribution continues to be undertaken by a few top contributors (Fayyaz et al., 2020; Taraszewski, 2017; Terjesen, 2003). Therefore, researchers recommend targeting *autonomous job motivation* which results from intrinsic factors such as joy and meaningfulness inherent in the work people do. This enables individuals to share knowledge voluntarily (Bidee et al., 2013; Frakes & Fox, 1995; Gagne et al., 2019; Karim & Majid, 2018).

Experts need trust and confidence in the accuracy of the extracted knowledge (Dong et al., 2015) – linking back to *JTB* explained in Section 2.4.1. Without mathematical verification techniques, they often prefer to be familiar with the knowledge creator before reusing located knowledge. “Knowledge sharing is a human interaction process that depends mostly on other human- and organisation-related factors” (Fayyaz et al., 2020, p. 11). While IT can support knowledge sharing, primarily it requires trust on the recipient side to absorb the knowledge input not invented here (NIH). A knowledge-sharing culture can boost a company’s innovation potential (Fayyaz et al., 2020).

Apart from the above-identified knowledge-sharing factors, there are three concepts that impact knowledge sharing: asymmetrical distribution of knowledge, asynchronous distribution of knowledge and push versus pull, which is discussed next.

Knowledge is distributed asymmetrically (Jasimuddin, 2005) when one party, the knowledge-rich, has access to a greater pool of knowledge than the other, the knowledge-poor (Clarkson et al., 2007). In the context of this research, a software asset owning team is knowledge-rich. Efficient knowledge storage and clever knowledge distribution allow knowledge sharing with potential reusers.

The sharing of previously stored knowledge can be asynchronous in nature, both in terms of time and location (Sahibzada et al., 2019). Here the knowledge storage acts like a buffer preserving knowledge (Ali et al., 2018).

There are two ways to distribute knowledge – via a *push or pull* mechanism. The pull approach requires employees to voluntarily locate, select and make use of previously

uploaded and maintained knowledge. Wallace (2018, p. 7) summarises his decades of experience by saying “uncontrolled development of pull- ... content, ... leads to all sorts of uncontrolled costs”. Unadministered, hoarded knowledge fosters creativity only by chance. While many researchers naturally assume knowledge is pulled out of repositories (Mäkitalo et al., 2020), practitioners recommend pushing selected knowledge in a planned way to the user is more effective (Wallace, 2018).

One reason for the strong management focus on knowledge creation, storage and distribution is that it is measurable. However, knowledge has to be applied to unfold its effects.

2.4.8 Knowledge Application and Reuse

Knowledge application aims to benefit from the created, stored and shared knowledge (Leber et al., 2015; Masa'deh et al., 2019). Knowledge application leads, via commercialisation, to financial benefit (Leber et al., 2015). Knowledge application enables innovation (covered in Section 2.5.6) when creating new knowledge.

Knowledge reuse is a special case, a subset, of knowledge application; it is the repeated application of knowledge (Merriam-Webster, 2021a). “Knowledge reuse is an important part of successful design in general” (Jonsson et al., 2021, p. 1053). Its effect depends on when the search for reusable knowledge begins, what the reuse conditions are and what knowledge finally gets reused (Jonsson et al., 2021).

Applying knowledge requires people to *self-assess existing knowledge* and respond to an internal or external *trigger* to reach out to wider knowledge (Fogg, 2009). Top commercial users of knowledge stores tend to be young professionals who spot a knowledge gap. However, only 5% of retrieved knowledge is applied, suggesting recipients do not persevere (Terjesen, 2003). The small group of experienced individuals benefitting from knowledge search and retrieval often do not reuse the knowledge directly but use it to identify key knowledge owners to talk to and key clients to present the knowledge to (Terjesen, 2003). Here knowledge applications quickly move away from a system-supported process towards a people-driven approach which is hard to track and even harder to measure and report on (Terjesen, 2003). The successful transfer of knowledge on reusable software assets can be achieved via a repository but is best enhanced by tacit knowledge via informal personal conversations.

Locating the required knowledge requires *effort and time*. Explicit knowledge is easier to locate and reuse. In contrast, localising implicit knowledge requires methods like storytelling and subsequent documentation (Kelleher, 2018).

Assessing and comparing knowledge against an actual need (Sandhu & Batth, 2021a) is often challenging since it is difficult to convey tacit knowledge (Nakamori, 2020). Burciu & Kicsi (2015, p. 10) state knowledge “exploitation requires extremely well qualified employees who, in addition, shall endeavour to permanently learn”. The reuser of the knowledge must pick up explicit knowledge, for example, by looking at software code and reading the code documentation, decode its basic information (see Figure 2.11) and create his personal vision and understanding of the knowledge. Doing so without personal interaction, based only on what is available in the knowledge sharing system, may be challenging, as Nakamori (2020) describes in detail, confirming Terjesen’s (2003) earlier findings at Accenture.

First mentioned by Cohen & Levinthal (1990), absorptive capacity measures the ability to reuse existing knowledge when solving a problem (Carayannis, 2012). It includes two components: prior knowledge that can be explored and transformed, and the intensity of effort (Human, 2020; Yoo et al., 2016). Absorptive capacity influences technological and organisational learning and general knowledge transfer, thus enabling competitiveness in the market.

The “not invented here” syndrome is used by individuals who are reluctant to absorb knowledge (Hejase et al., 2014). Potential ways of overcoming this measurable and therefore manageable syndrome are selecting team members open to knowledge sharing in both ways and facilitating expert talks. They generally aid the knowledge transfer and, thus, the reuse of knowledge (Antons et al., 2017; Weissenberger-Eibl & Hampel, 2021). Informal talks with the knowledge creator often have multiple positive effects. They grant the *approval* for reuse from a creator’s point of view. Having the chance to obtain further *help* in applying existing knowledge typically eases knowledge application and *increases the amount of reuse*.

There is a strong managerial interest in leveraging knowledge because it is a key resource in today’s business organisations (Eid & Nuhu, 2011). While often neglected, *reporting on the reuse* and *reuse experience feedback* provides helpful insights and turns the

knowledge reuse process into a closed loop. However, like reuse, knowledge application is hard to measure (Terjesen, 2003).

2.5 Business IT

Section 2.5 resumes the IT discussion from Section 2.1 and enriches it with the business aspects of intellectual capital, reuse and knowledge management identified in Sections 2.2 - 2.4.

2.5.1 Software and Services

Picking up the discussion on reuse from Section 2.3, asset reuse typically comprises *software variants* (Krüger & Berger, 2020) *containing different feature functions or extensions* that enable the reuse to happen. These different feature functions are depicted in Figure 2.16 e.g., by a green triangle or a yellow star.

There are different ways in which IT providers can bring their intangible output – software & services – to the client. In the form of traditional B2B IT services (I) where the client buys the intellectual property (see the top section of Figure 2.16), or via a software product (III) if the IP ownership is not passed on to the client (see the bottom section of Figure 2.16). Alternatively, in the form of a software asset licence (II) (see the middle section of Figure 2.16).

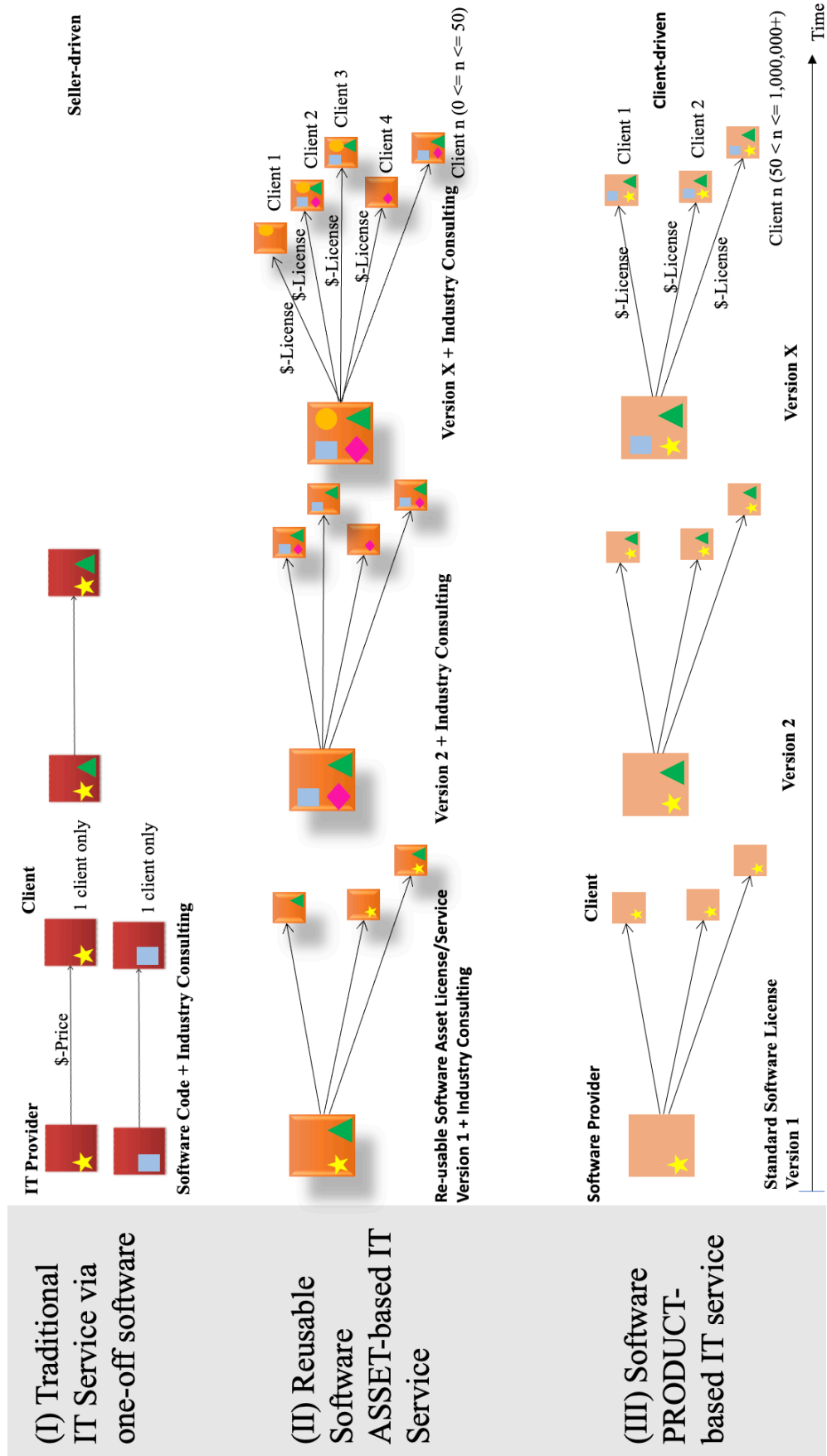


Figure 2.16 Tailored one-off IT-service (I) vs reusable software asset (II) vs software product (III) - adapted from (Wolfram et al., 2020)

Situation (I), the *traditional B2B IT service*, is typical for a B2B business when an IT provider creates software for a business client, perhaps complex, expensive banking software, custom-built for one specific commercial client who then fully owns it.

As can be seen at the top of Figure 2.16, if one client orders specific software with specific features indicated by the yellow star, this software remains unique even after an update, indicated by the green triangle symbolising an additional feature. If a second client needs software, they receive different, newly created software. The blue rectangle symbolises a feature different from the yellow star feature of the first client's software indicating that it has nothing in common with the previous software. In this construct (I) each client receives and owns an individual piece of software, including the right to decide later who can maintain, update, or expand the software via related IT services. The software is used, owned and controlled by the client like a commodity. It generates a one-time revenue for the IT provider.

The revenue an IT provider receives covers the change in ownership of the IP –meaning this software cannot be reused by the IT provider because the client owns it.

Situation (I) is often seller pushed and typical for B2B. Sellers provide consulting services to clients who, as a result, order tailored software.

The contrasting situation (III), *software product*, is when an IT company creates software that is of interest and value to many clients – either B2B clients or end users. Since software is non-rival (Haskel & Westlake, 2018), i.e., non-exhaustive, no mass production is required. Whoever owns the software can use and benefit from it. To generate revenue, ownership must remain with the IT provider.

A software product licence must be granted to a user to generate intellectual rents. Each client makes use of the same software features indicated by the yellow star in Figure 2.16. New version updates – indicated by the green rectangle in the lower section of Figure 2.16 – are triggered by the IT provider. Every user uses the same software with the same features, indicated by the yellow star and the green rectangle in Figure 2.16. This remains the case following any further software enhancements indicated by the light-blue rectangle in Figure 2.16. An example of this licensing situation (III) would be an iOS update or a Microsoft Office update. The user fully depends on the IT provider and is unable to suggest changes or modify the software (Baillon-Bachoc et al., 2021).

Situation III is often client-driven, representing a pull scenario whereby, for example, an end user or commercial user requests a software licence.

The middle of Figure 2.16 (II) depicts the reuse of a *software asset* – the central topic of this thesis. An IT provider offers a piece of software not just to one but to a few clients. However, the client set is not big. Typical examples of software assets could be foreign exchange trading software in banking or media management software for TV stations.

In situation (II), the IT provider creates software with a set of features, e.g., the yellow star and green triangle in Figure 2.16. However, the varying clients have particular needs meaning that not all of them would benefit from these two initial features. There may be clients who use only the green triangle and others who use only the yellow star feature. Over time, new clients may demand the addition of a further software feature (depicted by the pink rhombus in Figure 2.16). The IT provider can offer the new rhombus feature to his existing set of clients, and some may wish to adopt and pay the licence fee for it. As a further new feature (in the form of the orange circle) emerges, the software installed at the clients becomes diverse.

Different clients may parallel use the software asset - each with individually selected features. As in situation III at the bottom of Figure 2.16, the software II is owned by the IT provider. The user cannot modify it and fully depends on the IT provider when it comes to updates or maintenance. Thus, software created some time ago can offer the IT provider a licence/services revenue stream over a long time. In contrast to situation III, the IT provider who owns the asset faces challenges due to the diverse software installed at their clients in the market.

“Software embodies a rapidly growing part of our cultural, scientific, and technical knowledge” (Di Cosmo & Zacchiroli, 2017, p. 1). Software assets represent “special-purpose, intangible knowledge” (Baetjer, 2000, p. 150) – semantic and syntactic (Sodhi & Rattan, 2021), domain, information technology, design, project management (Devanbu et al., 1991) and business knowledge. Consequently, the knowledge management processes identified in Section 2.4.4 can be applied. A software asset can be created (Section 2.5.2), stored (Section 2.5.3), distributed (Section 2.5.4) and reused (Section 2.5.5). The following sections describe reusable software asset specifics going beyond the previous analysis of knowledge management. Section 2.5.6 explains the impact of reuse on innovation.

2.5.2 Reusable Software Asset Creation

Software asset creation is a specific form of knowledge creation discussed in Section 2.4.5. One of the challenges in the creation of reusable software assets is to ensure reusability (AL-Badareen, 2021), enabling software to be applied in multiple different situations as explained in Section 2.3.

Reusability results from software evolution and represents quality because the software has been previously tested and applied (Vizcaino, 2017). Technical rules that help software to be reusable include minimisation, durable design, versatile architecture, reversible design, transformable systems approach, modularisation, independence, robustness, standardisation, adaptability, extendibility, portability and maintainability (Fivet & Brütting, 2020; Kiran & Vindhya, 2011; Krueger, 1992; Mignacca et al., 2020; Sandhu & Batth, 2021a; Vasantha et al., 2020). Satisfying the reusability requirements is hard to quantify and does not guarantee effective reuse. Additionally, reusability metrics based on technical aspects like coupling, cohesion or branching depth (Ampatzoglou et al., 2018; Manjhi & Chaturvedi, 2019; Sandhu & Batth, 2021a; Zozas et al., 2019) have been proposed. But, still, no proven unit of reuse measure exists (Fivet & Brütting, 2020). Further, the software to be reused should be well documented, available, of manageable complexity and of good quality to avoid bugs being passed on during reuse (Zozas et al., 2019). Implementing reusability requires investment.

While the creation of a brand-new reusable software asset can be planned (1), its market success is more difficult to predict than that of a software product because it is targeting a long time period. Or, reusable software assets can emerge from (2) custom-made software (AL-Badareen, 2021; Mateen et al., 2017) (see Figure 2.17). In this case, reusability is implemented over time. Alternatively, software that has been reused before can be further reused and is listed either in an internal repository (3) or externally available (4) commercial-off-the-shelf (COTS) either for money or as open source (AL-Badareen, 2021; Mateen et al., 2017).

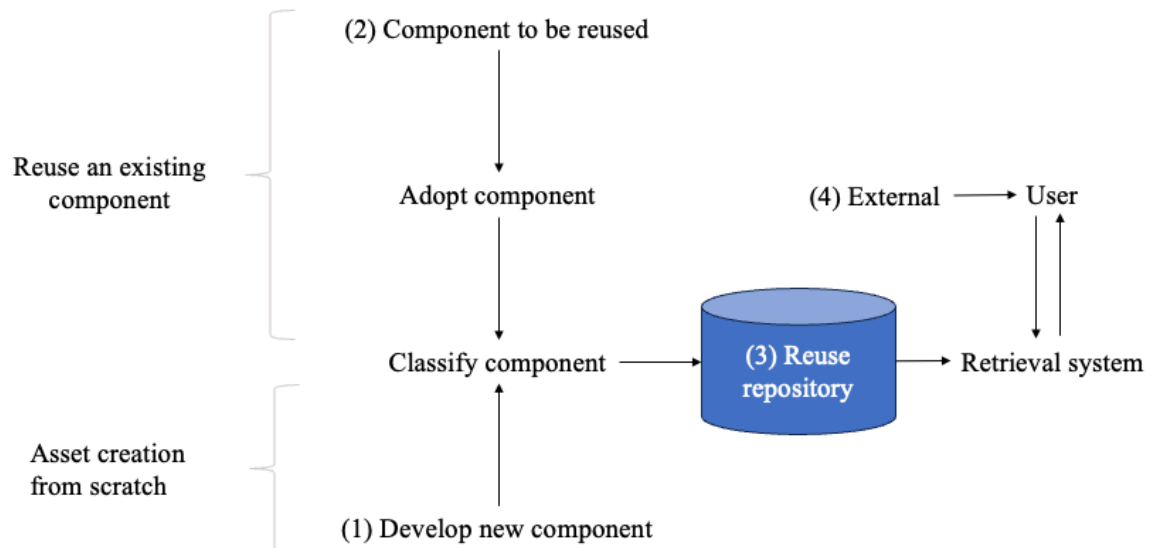


Figure 2.17 Four ways to obtain a reusable software asset for internal reuse - adapted from (Mateen et al., 2017)

It is vital prior to creating new individual knowledge that individuals are motivated to understand the present group knowledge. Fogg’s Behaviour Model (2009) lists three essential factors: motivation, ability and a trigger.

Motivational support needs to address the three core motivators Fogg (2009) identified: immediate pleasure versus pain, anticipated hope versus fear and social acceptance versus rejection. Hope is, out of these, the most potent and ethical motivator. Apart from motivation, simplicity is even more needed to “move users across the behaviour activation threshold” (Fogg, 2009, p. 5) caused by human laziness or inertia (Ahrne & Papakostas, 2001; Schumpeter, 2017). In the example of working on a new software project, this requires simple and easy access to past projects that could potentially contain reuse-relevant software. Fogg (2009, p. 6) states “Simplicity is a function of a person’s scarcest resource at the moment a behaviour is triggered”. Accordingly, an individual should be enabled to find past projects’ software, ideally quickly, at low cost, with low physical effort, little thinking effort, without damage to social status and as part of the daily routine.

Motivation and ability are prerequisites, but it takes what Fogg Field (2009) calls a trigger for a behaviour change to happen. Dreyer & Wynn (2016) propose triggers (see Figure 2.13). Triggers act in multiple ways, e.g., as behaviour-motivating sparks, behaviour-

easing facilitators, or as reminding signals (Toxboe, 2021). Here, in this context, behaviour-easing would be sharing a list of past software projects along with expert names at the right time to motivate teams to consider existing software assets for reuse.

Jarvis (2009) describes how Google develops software jointly in partnership with the client. This entails receiving requirements directly from the client, resulting in a software solution that precisely meets the client's needs. By working this closely with the client, IT providers gain deep insight into the business of their clients. The business success of Amazon is partially based on the deep knowledge Amazon has of its clients (Jarvis, 2009).

The development of software has moved from a waterfall principle towards the agile software development model (Behutiye et al., 2022; Zorzetti et al., 2022). This increases the extensibility and therefore reusability of applications (Jarikre et al., 2022). The agile approach is often applied as an entrepreneurial methodology in start-ups that continuously experiment with product management and may develop a business plan iteratively after evaluating business success (Zorzetti et al., 2022).

During the creation process intellectual property can undergo a commercialisation activity “by which the knowledge, ideas and inventions generated by the research and development are converted into the business assets capable of producing commercial revenues from marketable goods and services” (Rider et al., 2006, p. 1). The literature mentions surprisingly little about the commercialisation of software assets.

2.5.3 Reusable Software Asset Storage

Software storage represents a specific form of knowledge storage as discussed in Section 2.4.6. However, since only explicit knowledge can be stored, according to Figure 2.14, the tacit and implicit knowledge remains excluded and therefore missing (Spender, 2008). Further, since its build-up is a personal accomplishment depending on the information input (Oppenheim et al., 2003) and other factors, different users interpret stored information differently. If changes are applied, repositories must be kept up to date.

As Sandhu & Batth (2021a) explain, storing reusable software assets requires *abstraction* (Biggerstaff, 1994) to identify relevant asset characteristics, e.g. use cases, and classification to store assets according to these characteristics in asset repositories. Burciu & Kicsi (2015) describe how *abstraction* is generally required at the point when information turns into knowledge.

Depending on the purpose and the audience, software knowledge can either be stored as *software code* via GitHub (El Baff et al., 2021; Hasselbring et al., 2020), relevant for developers, or as *meta-data* along with relevant artefacts via knowledge repositories, relevant for sellers and solutioning teams (Abdelwhab Ali et al., 2019; Arvanitou et al., 2021; Sahibzada et al., 2019).

Di Cosmo & Zacchiroli (2017) recommend *storing source code* in all its versions in order to understand developers' ideas, including all changes and updates, fully. They implicitly assume it is the reuser who modifies the code.

AL-Badareen (2021) mentions transition operations that apply after the creation of reusable software assets. These activities include cataloguing software assets, searching for software assets, locating a software asset in the catalogue or in the market. What he describes, without naming and further detailing it, is a *software asset meta catalogue* where information about reusable software assets is stored for future use.

In practice, teams hesitate to prepare information for future reuse (Jonsson et al., 2021). "Even where engineers do trust the systems, they are very often not willing to spend significant time organizing documents to assist their future access, owing to the difficulty of decision-making in this task" (McMahon et al., 2004, p. 313).

Proposed reuse library metrics include time in use and number of reuses (Imoize et al., 2019; Ram & Devi, 2019). Measuring the number of stored assets is simple. However, since it purely reflects the quantity of assets, little can be said about the quality, future use and business impact of a reusable software asset.

2.5.4 Reusable Software Asset Distribution

Software assets are created by individuals who then form an asset-owning team. Creating and storing asset information is not sufficient. The organisation must be informed to gain knowledge about the asset - that is what is meant by asset distribution.

Asset distribution is a specific form of knowledge distribution discussed in Section 2.4.7. It can take place via e-mail campaigns, education sessions, or mandatory processes driving teams to make use of the reusable software assets available in the asset repository. The current literature provides no guidance on the strategy of how, when and with whom to share news on the existence or enhancement of a reusable software asset.

To spread the word about the existence of reusable software assets to a wider audience of reusers, the above-mentioned software asset meta catalogue is required. It acts like a sharing platform in the *sharing economy business model*. As such, debates are ongoing about the degree of collaboration, the definition of sharing, and the logic of exchange (Rose, 2021). As in Figure 2.18, the platform/asset repository can act as a collaboration/distribution tool while the handover of the asset from the owner to the seeker/reuser can be platform independent.

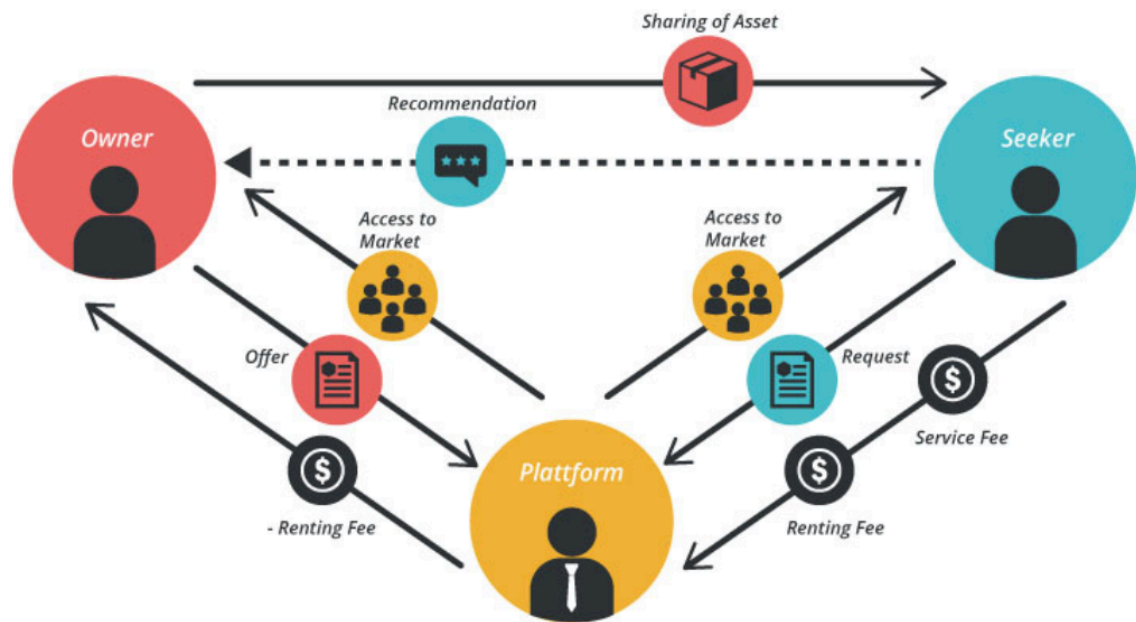


Figure 2.18 The sharing economy business model (Beha et al., 2022)

Reusable software assets often target a *specific industry or domain*. To increase the chances of vertical reuse of the software asset, it makes sense to share information about the asset within this specific industry or domain, as it provides the highest chances of reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010).

Knowledge work is hard work. Due to *personal inertia* (Ahrne & Papakostas, 2001; Lam, 1997; Ormerod, 2020) humans are reluctant to acquire new knowledge. Comprehending reusable software assets created by others and comparing them against current requirements is initially a bigger task than starting a new development from scratch. Knowledge is better pushed to the reuser rather than pulled (Wallace, 2018). This has effects on the asset-sharing concept addressing the trigger in Fogg's Behaviour Model (2009).

Jarvis (2009) mentions the importance of sharing new software features in a way that can simply be googled, both by people and search machines. According to him, software features should be made available in such a way that all potential questions a client has would be answered.

2.5.5 Reusable Software Asset Reuse

The list of software-related artefacts that can be stored and reused is long and includes code, offerings, tools, accelerators, solutions, sales material, documentation, system specifications, estimates, templates, plans, test cases and data, business knowledge, requirements, hardware or any combination of these (AL-Badareen, 2021; Bauer, 2016; Frakes & Terry, 1996; Frakes & Fox, 1995; Irshad et al., 2018; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Spoelstra et al., 2011; Woerner et al., 2013; Yoshida, 2014) (see Figure 2.19). Following one of KPMG’s (2018, p. 15) five reuse principles, “Try to reuse ... as a whole, if that is not possible as parts ...”, it is best to reuse software code and its *reusable artefacts*.

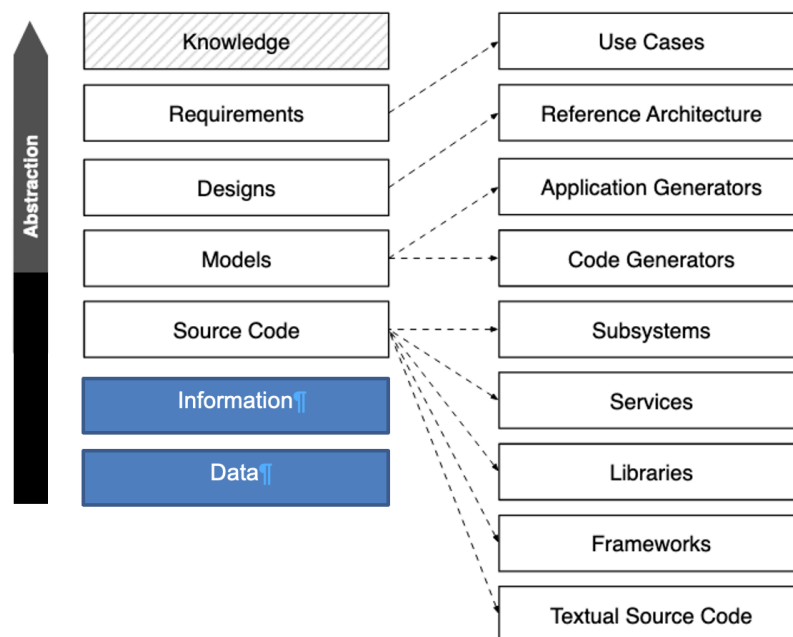


Figure 2.19 Examples of reusable software-related artefacts - adapted from (Bauer, 2016)

A plug-and-play theory assumes a Commercial-off-the-Shelf (COTS), open source or similar item (Anderson et al., 2007) can be used again and again, like taking a spare part off a shelf. In line with Zhu (2009) and others, AL-Badareen (2021, p. 20) describes reuse as “copying software asset from a legacy system and use it directly in the development of new software system”. Figure 2.20 visualises this concept by showing how the asset A,

in blue, resulting from the legacy system X is getting reused in projects Y and Z in grey. However, this approach does not consider the different experimental systems and the different item variants that Cacioppo (2015) and Schloss (2018) identified (see Section 2.3.1).

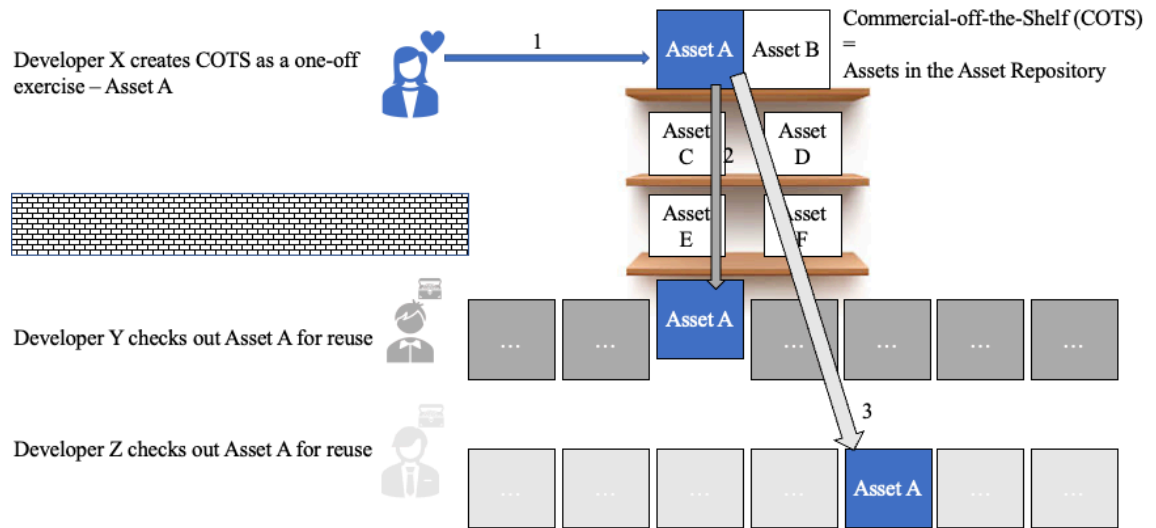


Figure 2.20 Theoretical software asset reuse scenario (source: author)

On the other hand, AL-Badareen (2021, p. 20) explains reuse “is a process of modifying software asset in order to achieve the new requirements of the new software system”. This statement is in line with “The pragmatic reuse of software revolves around the invasive adaptation and reuse of code for purposes for which it was (probably) not originally intended” (Kessel & Atkinson, 2015, p. 64) and what is earlier said in Section 2.3.1.

Lear (2021), who researched video game assets, states that following the adaptations required for reuse, the video asset itself is constantly changing from a source asset to a final asset, thus confirming the dynamic capability view (Marrucci et al., 2022). This way, each reuse can be regarded as a mini release of a super-flexible product, the reusable software asset. Adaptive reuse is a kind of sustainable renewal as it prolongs the life of the asset (Abdulameer & Sati’Abbas, 2020; Lear, 2021; Mihale-Wilson et al., 2021; Whalen et al., 2018).

AL-Badareen (2021) does not state who would be performing the software adaptations. Whenever the literature specifies an actor, it is the reuser acting (Kessel & Atkinson, 2015; Mäkitalo et al., 2020). AL-Badareen (2021) suggests conducting a case study to

analyse the applicability of his transformation and transition ideas. This is where this thesis picks up.

Vertical (inner-domain) and horizontal (cross-domain) reuse can be differentiated (Anasuodei & Ojekudo, 2021; Jalender et al., 2010).

While much has been written about reuse, little is known about how to get reuse started. It can be assumed that individuals prefer the conventional process of ‘going for new’ if they require a commodity or software (see the white box on the right in Figure 2.21).

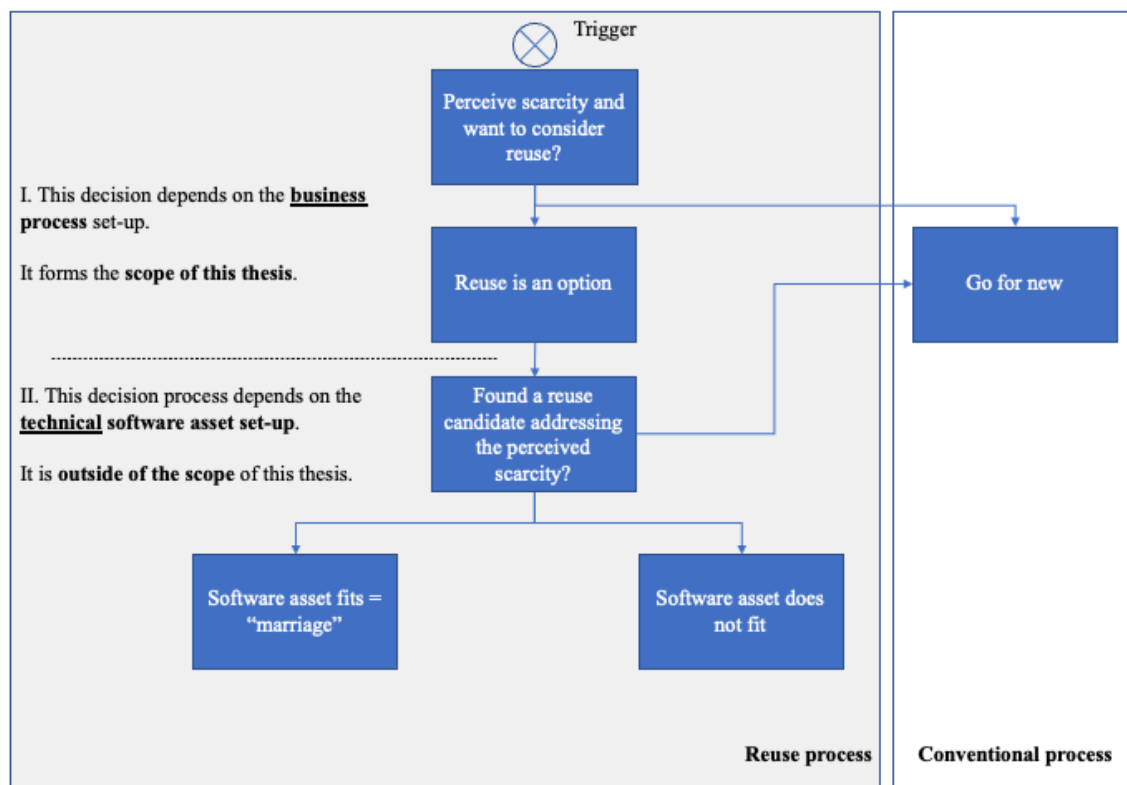


Figure 2.21 Considering an asset for reuse (source: author)

The first step towards reuse is when an individual actively considers reuse as an option - see (I) at the top of the grey depicted “Reuse Process” in Figure 2.21. According to Fogg’s Behaviour Model (2009), it takes three essential factors to change behaviour: motivation, ability and a trigger.

Yeh et al. (2006) identified strategy and leadership, culture, people and technology as critical success factors for achieving organisational effectiveness. While these concepts help to motivate and enable teams, they miss Fogg’s (2009) trigger.

Dreyer & Wynn (2016) introduced the concept of a *reuse trigger* to encourage individuals to break out of their normal habits during knowledge acquisition - see Figure 2.13 in Section 2.4.3. Subsequently, a reuse-stipulating trigger is required during software asset reuse – a fact not clearly addressed in software reuse literature. Scarcity triggers reuse (Fivet & Brütting, 2020; Salgot et al., 2017; Stahel & MacArthur, 2019) but also other pre-set reasons or conditions can cause reuse (Jonsson et al., 2021). For example uncertainty due to high complexity or a big choice can trigger reuse (Jonsson et al., 2021).

For cases where no obvious trigger applies, Fraser (2021) describes how he tried to push software reuse using technology through a “workshop process to help teams identify opportunities for software reuse”. By regularly asking a “trigger question” stipulating reuse, he achieved proven reuse success (Fraser, 2021, p. 5). Triggers can provide a mechanism for initiating software asset reuse (Majchrzak et al., 2001; Rose, 2019).

The phase II in Figure 2.21, covering locating, selecting, reusability assessing and integrating the software asset by using different reuse strategies (AL-Badareen, 2021; Hofstetter, 2009; Krüger & Berger, 2020; Lange, 2020; Sandhu & Bath, 2021a, 2021b), has been described in detail in IT literature and is beyond the scope of this thesis.

Typically experts with more experience have a higher reuse rate (Jonsson et al., 2021; Terjesen, 2003). “If the designer is not aware that a tool that could be interesting exists, reuse is not going to happen.” (Jonsson et al., 2021, p. 1061). Experts have seen more solutions and know what exists. From this it can be concluded that to initiate reuse, existing knowledge needs to be located and understood (Jonsson et al., 2021).

Trustworthiness is an important prerequisite of reuse. Teams only want to reuse knowledge that has been proven to work well in the past or requires only a few iterations of testing and revision (Jonsson et al., 2021). “Unless they already knew if the tool had been working well, they would check with other[s] ... before reusing solutions” (Jonsson et al., 2021, p. 1061).

Mihale-Wilson (2021) differentiates between an opportunistic and strategic software reuse principle. Opportunistic reuse refers to ad-hoc, unplanned reuse. Strategic reuse describes *systematic and controlled reuse* (Mohagheghi & Conradi, 2007). It typically requires upfront investments but increases the chances of success (Fortune & Valerdi, 2013).

Reuse is an ability an organisation has to acquire. An organisation's ability depends on its capabilities and the maturity achieved within these capabilities (Succar, 2013). Holibaugh et al. (1989) first issued a reuse maturity model as "an aid for performing planning and self-assessment to improve an organisation's capability to reuse existing software artefacts" (Tripathy & Naik, 2014, p. 334).

The best possible reuse, and therefore efficiency gain, can be achieved by systematic reuse. While reuse maturity models list systematic reuse as the highest achievable maturity level, they do not provide details on how to achieve systematic reuse. Fivet et al. (2020) regret the lack of reusability measures.

Reuse often requires *pushing an existing software asset into a project* (Wallace, 2018). 'Marrying' the asset with the project at the right point in time is a critical task. It takes both the asset owner and the reuser to agree to the reuse.

2.5.6 The Impact of Reuse on Innovation

In the 5th century BC, the Greek philosopher Anaxagoras said: "Nothing is born or perishes, but already existing things combine and then separate again" (Ritter, 1838). This is one of the earliest available statements on reuse: combining existing ideas in new ways or extracting parts to integrate them to form a new creation (Flath et al., 2017; Majchrzak et al., 2004; Markus, 2001).

Schumpeter (2017) defines a new *combination of existing resources* as innovation. Stenholm (2019, p. 141) warns "reuse strategies can be counterproductive to innovation since they favour incremental development of existing technologies". However, it depends on how different the new experimental system is, in which components are verified or extended so that their combination forms a new, more innovative component or piece of knowledge (Cacioppo et al., 2015; Horbach & Rammer, 2020; Schloss, 2018). The more different the experimental system, the more innovative the combination.

Later, Schumpeter redefines innovation by positioning it as the *initial commercialisation* of an idea, setting it further apart from an invention anyone can achieve everywhere (Fagerberg, 2003). In short, "innovation is the transformation of an invention into practical application" (Baptista et al., 2015, p. 59). Tidd and Bessant (2020) detail Schumpeter's commercialisation concept by saying innovation is all about opening up

new markets. While innovation has many facets, it is a positive driving force that organisations are keen to achieve.

There are different types of innovation outcomes such as product innovation leading to effectiveness, production methods innovation leading to efficiency, service innovation, marketing innovation, exploitation of new markets, business model innovation, organisational innovation or supply chain innovation (Kahn, 2018). Software reuse can help achieving these kinds of innovation.

While innovation is targeted, because of its competitive effect in the market, it cannot be planned for. Innovation is based on the emergence of a new thought leading to a radical change (Leber et al., 2015). Since systematic reuse allows people to operate at a higher knowledge level, the creative process of innovation is more often, more systematically going to happen. Dynamic change capabilities provide the foundation for innovation – substantiating the Dynamic Capabilities Theory (Marrucci et al., 2022).

2.6 Objective of this Research and Research Questions

This thesis aims to identify critical knowledge processes that enable the systematic reuse of software assets developed in and owned by a global IT enterprise. This is achieved by revisiting and sharpening the term reusable software asset, investigating the available definitions of systematic reuse with respect to their practical application in a global IT company and identifying how software asset reuse can be systematically enabled from a knowledge process perspective.

Global IT enterprises provide software solutions to clients from various industries. The IT business is a growing market impacting the whole of society in all aspects, such as business, culture, and politics (Roztocki et al., 2019). One hugely advantageous trend in the IT environment is globalisation, giving access to IT solutions worldwide. One disadvantageous trend, however, is the ongoing push for low-cost business. A company with a systematic handle on asset reuse could be able to answer a request for proposal (RfP) ahead of competitors providing a proven, low-risk solution that can immediately be demonstrated to the client.

Analysing expert interviews, this case study research answers three research questions.

RQ1: How can a reusable software asset be characterised?

RQ2: How can the systematic reuse of a software asset in a global IT enterprise be specified?

RQ3: How should the knowledge-related processes critical to enabling systematic software asset reuse in a global IT enterprise be shaped?

This research focuses on a global IT company because it faces specific challenges, such as a distributed workforce with employees not necessarily known to one another. At the same time, a global IT company operating in many locations with thousands of employees is dealing with a sufficiently large set of repetitive business problems that can be addressed by software asset reuse.

2.7 Theoretical Framework

This thesis researches the *phenomenon* of systematic software asset reuse. The theory behind this is multidisciplinary, encompassing various concepts, principles and best practices from intellectual property management (Section 2.2), business IT management (Section 2.5), knowledge management (Section 2.4), as well as leadership and business management (Sections 2.4 - 2.5). The theoretical framework to be established for the systematic reuse of software assets aims at identifying *key variables* within critical knowledge management processes that enable systematic software asset reuse.

To identify the key variables that influence the phenomenon under research and to avoid solution jumping, the research scope needs to be expanded (Enders et al., 2016) by looking at processes that are adjacent to the technical reuse in IT.

Researchers demonstrated that leadership affects intellectual capital, as detailed earlier in Sections 2.4 and 2.5. Research relates transactional leadership with the creation of knowledge, whereas the storage and distribution of knowledge is rather related to transformational leadership. Despite a trend towards transformational leadership, this can make sense because the creation of software assets is, contrasting to storage and distribution, rather countable, allowing goal setting. It can be claimed that supportive leaders are a prerequisite for generating intellectual capital such as reusable software assets.

Further, intellectual capital has a direct impact on innovation which, in turn, is related to the organisational performance (Alrowwad & Abualoush, 2020; Fayyaz et al., 2020; Obeidat et al., 2021) as shown earlier in Figure 2.2.

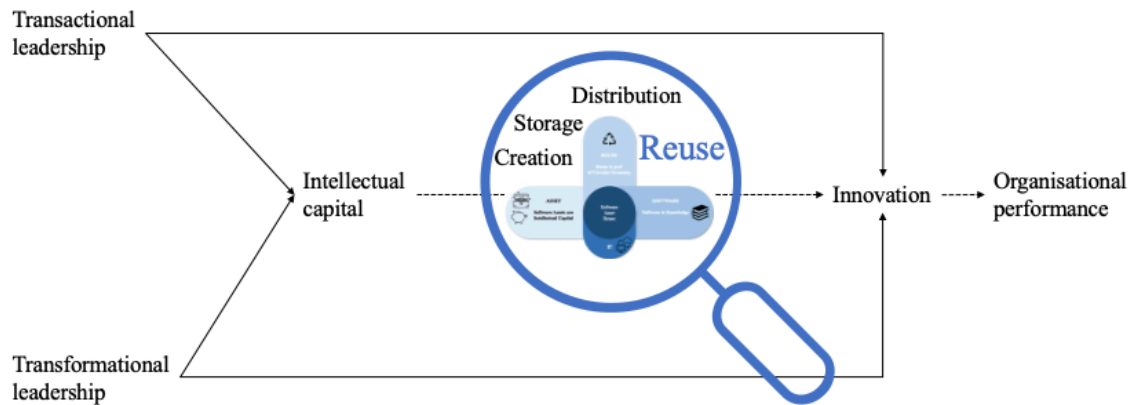


Figure 2.22 Intellectual capital directly impacts innovation - adapted from (Alrowwad & Abualoush, 2020)

Figure 2.22 depicts the theoretical framework for this thesis. Reusable software assets can be classified as intellectual capital (see Section 2.1). At the same time, reusable software assets represent knowledge (Di Cosmo & Zacchiroli, 2017; Dreyer & Wynn, 2016; Huang, 2019; Kneuper, 2002). If well managed, knowledge can drive innovation (Majchrzak et al., 2004) which is a key determinant for organisational success. This thesis highlights two specific ways of organisational performance – revenue and savings as described in Section 2.2.

This thesis investigates the interplay of intellectual capital, knowledge management and reuse as part of the circular economy. Its findings detail the gap between intellectual capital and innovation in Figure 2.22 via the managerial guide in Section 4.8. This can be used as input to set the future agenda for transformative and transactional leadership.

Knowledge can be managed according to four key knowledge processes: creation, storage, distribution and application, as shown in Figure 2.22 and explained in Sections 2.4.5–2.4.8. Consequently, this thesis applies these four well-researched knowledge processes to reusable software assets to identify *key variables* that aid software asset creation, storage, distribution and, finally software asset reuse. The objective is to identify how these key processes *differ* and *under what conditions*. This is a justifiable approach since reuse has both non-IT and IT aspects (see Figure 2.21). The non-IT aspect of

software asset reuse, pure knowledge management, is the prerequisite for reaching the IT-stage and forms the key scope of this thesis.

Investigating what is required to enable software asset reuse from a knowledge management perspective allows the unusual economic behaviour of reusable software assets to be identified and considered and provides a more realistic view of the niche area of software asset reuse in today's B2B business.

This thesis aims to provide a reuse management contribution to knowledge and practice. Based on gained interview data a generalisable theory on software asset reuse can be, within limits, inductively formulated bottom-up - as a contribution to knowledge. It allows to deductively draw top-down specific conclusions and make predictions, summarised in a managerial guide, that contribute to practice.

3. Research Methodology

This chapter presents the methodological background to bridge the research gap of missing documented real-life reuse cases. It explains the author’s philosophical paradigm in Section 3.1, the research design (see Figure 3.1) in Section 3.2 and the case study planning, including quality assurance in Section 3.3. The chapter concludes by commenting on the research ethics in Section 3.4.

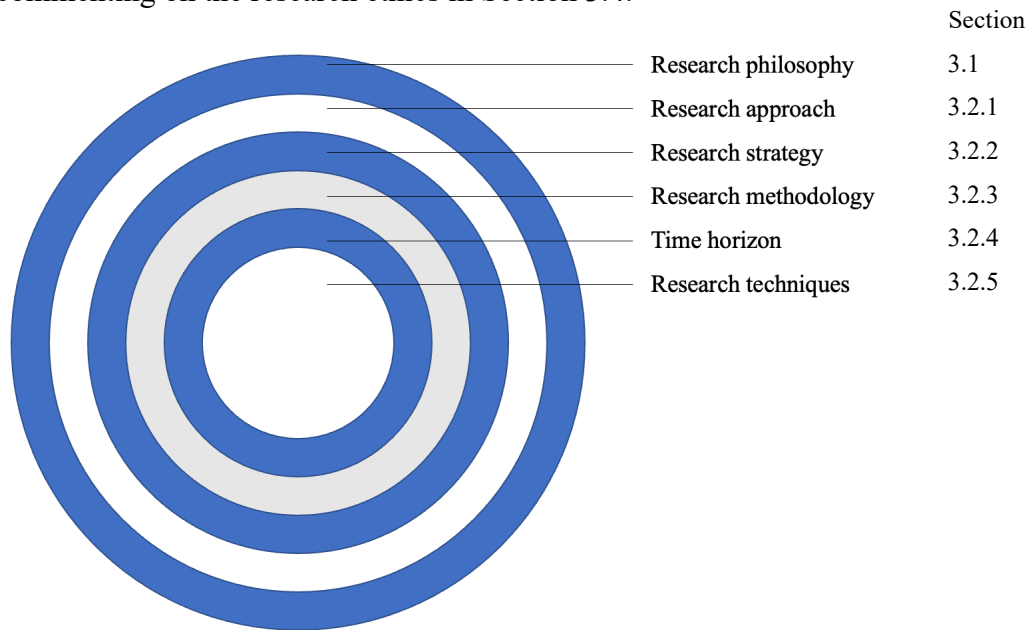


Figure 3.1 Saunders’ research onion simplified - adapted from (Saunders et al., 2019, Fig. 4.1)

3.1 Research Philosophy

The way knowledge is viewed has always been shaped by the debates of prevalent philosophers (Wickramasinghe et al., 2005) as well as by the society we live in (Ruben, 1979).

Ancient Greek philosophers discussed the importance of truth in knowledge. According to Plato, tacit and inarticulate knowledge may be key driving forces as they reflect the human character more than the right beliefs. This is important because the researcher’s philosophy may impact the research results. Following Aristotle, even objective knowledge is value-loaded (Chappell, 2012).

Table 3.1 lists philosophical views on knowledge. Pragmatism forms the basis of the author’s thinking. It is the underlying philosophical methodology to conduct this research

on software asset reuse. Pragmatism values the practical application of knowledge and understanding. Following this, truth results from the knowledge of a local reality based on our experience (Nowell, 2015) - see Table 3.1, third row. Subsequently, it is an appropriate philosophy for knowledge management case study research on the practical implementation and management of software reuse in an IT organisation. Being a trained engineer, the author of this research prefers to look at philosophical topics in terms of practical implications (Ormerod, 2020). She agrees with William James (1907) and Richard Rorty (1982), who believe that the crucial role of knowledge is to solve problems creatively. Additionally, pragmatism underscores the importance of understanding the context and situation, which is critical to acknowledging the unique challenges and opportunities organisations face when reusing software. The central research question, “How should the knowledge-related processes critical to enabling systematic software asset reuse in a global IT enterprise be shaped?”, is a complex business challenge in the social sciences field that emerged a few years ago. Conversely, interpretivism, which emphasises the subjective and interpretative nature of understanding (Goldkuhl, 2012), is deemed less appropriate for this study as it may not provide a reality-based and practical understanding of software reuse problems.

Table 3.1 Philosophical views on knowledge - adapted from (Lai, 1981; Wickramasinghe et al., 2005)

School of Thought	Basic Ideas on Knowledge	Some Proponents
Empiricism	Knowledge can be created from experiments and thus only mathematics and natural sciences can provide secure knowledge	Locke, Russel
Positivism	Knowledge is gained from the observation of objective reality	Compte
Pragmatism	Knowledge, representing a local reality based on our experiences, is seen as useful for action	Dewey, Peirce, James, Putnam
Constructivism	Knowledge is constructed in our minds, thus is not objective	Erlangen School
Interpretivism	Knowledge is socially constructed and subjective; it allows understanding	Weber
Sociology of knowledge	Knowledge is a socially constructed reality	Mannheim, Scheler
Critical theory	Uses knowledge to integrate the tension between reality of society and the real societal function of science	Habermas, Horkheimer
Critical rationalism	All knowledge must be open to empirical falsification before it can be accepted	Popper

Table 3.1 lists further schools of thought, like positivism or constructivism. They are observational and less objective than pragmatism; hence they are less applicable to the author's thinking. Critical schools of thought focus on critical knowledge, in the case of empiricism assuming knowledge results purely from mathematics. The researcher does

not believe in the sociology of knowledge (Mulkay, 2014). She regards knowledge as a scientifically constructed reality.

3.2 Research Design

3.2.1 Research Approach

This case study research takes an *abductive approach* (Eisenhardt, 2020; Khisty, 2000) by investigating reuse processes for relevant software asset reuse cases and identifying similarities and differences per case. Pragmatist Peirce introduces abductive reasoning, which Khisty (2000, p. 116) describes as “... a learning process ... able to isolate and tackle wicked problems”. Abductive reasoning relies on creative problem-solving and innovation by the community via a democratic process (Ormerod, 2020). In practical terms, the author abductively reasons, based on real-life but incomplete experience, to best predict a practical process for systematic software asset reuse.

Abduction is appropriate for this study as it allows the integration of deductive and inductive methods. In inductive reasoning, data from specific observations are closely examined to derive general conclusions. In deductive reasoning, general rules are used as theoretical statements to derive specific conclusions (Yin, 2018). Subsequently, abductive reasoning enables the development of new hypotheses and predictive theories. It makes it possible to question existing theories and hypotheses critically and to identify new fields of research and exploration (Eisenhardt, 2020). In addition, the abductive approach considers a variety of perspectives and data, develops a more complete understanding and compensates for incomplete observations.

The desired theory on the ideal software asset reuse process results from combining the results of the literature review and practice. This study aims to locate and expand existing theories on captured reuse cases in addition to responding to rival theories on software asset reuse. That interaction leads to a more complete research design (Rosenbaum, 2002). The applied generic framework offers recommendations and guidance for the practical implementation of systematic software asset reuse.

3.2.2 Research Strategy

There is a fundamental choice between quantitative research design and qualitative research design (MacDonald & Headlam, 2008), or a combination of both via mixed methods research design as shown in Table 3.2.

Qualitative research typically investigates complex processes (Aspers & Corte, 2019; Atieno, 2009). This fits well since this thesis investigates the processes enabling systematic software asset reuse. Second, it interprets the interviewees' insights and experiences. This is useful because the exact mechanisms of what triggers the reuse of a reusable software asset have not yet been fully uncovered. Third, a qualitative researcher can be seen as the primary data collection and analysis instrument. Fourth, qualitative research involves fieldwork in the way that the researcher communicates with experts in the field. Finally, it is descriptive in nature using words or graphics to describe the research relationships which match the targeted outcome of this study.

In contrast, quantitative research tends to confirm patterns that focus on numerical responses rather than context. It helps answer "how much"-questions, while this research aims to answer "how"-questions (Yin, 2018). The researcher regards the area of software asset reuse as not yet well understood to apply quantitative research.

Table 3.2 Research methodologies - adapted from (MacDonald & Headlam, 2008)

Research Methodology	Mixed	
	Quantitative	Qualitative
Aim	Count things to explain what is observed	Detailed description of what is observed
Purpose	Generalisability, prediction, causal explanations	Contextualisation, interpretation, understanding perspectives
Tools	Researcher uses, e.g., surveys as tool to collect numerical data	Researcher is the gathering instrument
Data collection	Structured	Unstructured
Research strategy	E.g., surveys, experiments, longitudinal studies	E.g., case study, focus groups, observation
Output	Data in the form of numbers and statistics	Data in the form of words, pictures or objects
Sample	Large number of cases representing the population in focus. Randomly selected respondents	Small number of non-representative cases. Respondents selected on their experience
Objective/Subjective	Seeks precise measurement and analysis	Subjective – individual's interpretation of events is important
Researcher role	Researcher tends to remain objectively separated from the subject matter	Researcher tends to become subjectively immersed in the subject matter
Analysis	Statistical	Interpretive

While various qualitative research strategies exist, action research or case study research are the two most relevant to the research in hand. As a results-oriented pragmatist, the author was drawn to action research. However, the action research method (Coghlan, 2019; Dick, 1993; Eady et al., 2015; Johnson, 2020; Pitt, 2007) involves identifying an action response to a real-time change action in the chosen scenario. This seemed

challenging for the subject of this research, a global IT enterprise which is permanently transitioning. Pitt's (2007) Australian red-meat-industry action research study implicitly confirmed that "one of the weaknesses of action research is its localism and the difficulty ... in intervening in large-scale ... change efforts" (Brydon-Miller et al., 2003).

After discarding action research, the author of this study turned to an alternative research strategy, *qualitative case study* research (Eisenhardt, 1989; Lindgreen et al., 2021; Yin, 2018). By documenting real software reuse cases, the present study provides rich qualitative input detailing the context (Matallo, 2021) in a way that reuse situations have not been described before. This level of detail is relevant for deriving patterns and theories of successful software asset reuse during subsequent research. The way the case study data are collected fits with the business conduct guidelines of the chosen global IT company.

Any case study is a trade-off between the real world and the abstraction inherent in describing the cases (Runeson & Höst, 2009). The real world is too complex for one case study. A case study can be flexibly designed by adapting research parameters during the research process. Increasing the control by case simplification can lead to meaningless results. Thus, this author, from her pragmatist perspective, based on literature research and practical experience, has prepared a comprehensive set of interview questions to reflect the asset and reuse situation as closely as possible.

3.2.3 Research Methodology

There is a choice of mono-, multi- and mixed-methods research design. Mono-method designs apply a single data collection method. Multi-method designs rely on multiple methods – all either quantitative or qualitative. The integration of qualitative and quantitative methods is referred to as mixed-method design.

This thesis uses a pure case study strategy - a mono-method *qualitative* research methodology. This is based on the desire to gain a thorough understanding of the specific context and phenomenon being studied. The mono-method research provides a focused and deep investigation of a selected case to comprehend the factors influencing the systematic reuse of real-life software assets in line with the pragmatist philosophy of the researcher. Although a multi- or mixed-method research might have gained a broader perspective, they were dropped. First, due to concerns about justifying additional

interviewee effort. Second, because of the ability to effectively compare and analyse data across multiple contexts. Finally, the researcher regards the area of software asset reuse as not yet well understood enough to apply quantitative research e.g., via surveys. This rules out mixed methods research (Creswell, 2007).

3.2.4 Time Horizon

A research project can have two basic time horizons – either it is cross-sectional or longitudinal (Saunders et al., 2019). For this research, the *cross-sectional* time horizon applies “as the primary data collection occurs at a single point of time” (Saunders et al., 2019, p. 66) over a short period of a few weeks. Other research strategies, like action research, follow a longitudinal time horizon which involves collecting data covering a longer time span to identify changes over time. In contrast, this thesis focuses on a cross-sectional, current set of data to reveal patterns that support the systematic reuse of software assets.

3.2.5 Research Techniques for Data Collection and Analysis

Of the various data collection methods available for qualitative research, e.g., questionnaires, interviews or observations, the researcher deems the *interview* to be the best-suited data collection method for this thesis. Interviews provide data that, after data analysis, can be drawn upon to answer the set research questions. If the right questions are asked, interviews allow the behaviour, beliefs, rationale, understanding, thoughts and feelings that guide and directly impact the asset reuse work of the interviewees to be captured (Runeson & Höst, 2009; Stuckey, 2013).

The interviews were conducted with a selected set of subject matter experts. Synchronous online interaction was chosen to capture each interviewee’s individual point of view (Archibald et al., 2019). A multi-party synchronous online interaction could have been a possible alternative, perhaps in the form of an online focus group, but this approach was skipped: Some assets or their clients are subject to confidentiality, so related information cannot be shared. Discussions with multiple interviewees are typically dominated by a few, preventing others from talking. This is avoided by conducting individual interviews. They followed preparation sessions to locate and identify the most knowledgeable and talkative interviewees.

Generally, there are three interview types: structured, semi-structured and unstructured. Open-ended questions promise the richest data outcome as they can elicit answers the interviewer did not conceive (Stuckey, 2013). The researcher decided to go for a *semi-structured interview*.

Firstly, it allows for a more consistent and standardised method of data collection, as pre-set questions are used to guide the interview. This helps ensure that all participants are asked comparable questions, with an emphasis on aspects that a respondent could not address easily in a purely narrative interview (Kallio et al., 2016; Stuckey, 2013). It provides a more accurate and systematic analysis of the responses than an unstructured interview, as the questions give a clear framework for coding and categorising the responses.

Additionally, the semi-structured interview offers some flexibility in the interview process by avoiding forcing the interviewee into a strict interview regime, such as the structured more survey-type interview (Yin, 2018). Participants are encouraged to share their experiences in their own words. This can lead to a deeper understanding of the topic and reveal important nuances and insights.

Finally, the preferred semi-structured type of interview suits the personality, the business work style and the research plan of the interviewees and the interviewing researcher. It allows interviewing busy businesspeople within a predictable timeframe.

The pre-defined set of questions results from findings in the literature review (Kallio et al., 2016) as well as the author's work experience.

The research data collected must be analysed. Common data analysis methods are qualitative content or narrative analysis, interpretive phenomenological analysis, discourse analysis and grounded theory (Warren, 2020; Yin, 2018). Given the chosen semi-structured interview data collection method, the researcher regards *qualitative content analysis* as the most suitable data analysis method for this research. It fits the chosen data collection method best. All other methods focus either on the subjective feelings of individuals or examine rival explanations or pattern recognition – a type of analysis that was applied in a second step, after the targeted qualitative analysis, as a separate step of research.

There are three basic types of content analysis: conventional content analysis, directed, or summative content analysis (Hsieh & Shannon, 2005). All of them interpret meaning from the interviewee's answers which represent the data content input. The researcher decided to go for conventional content analysis i.e., the codes are directly identified from the input, independent of any theory or frequency. This way, the researcher can notice newly emerging research approaches, even if articulated by one or a few interviewees only. *Thematic analysis* (Braun & Clarke, 2006) is applied to implement the conventional content analysis. Directed content analysis is not applicable as it aims at validating an existing theoretical framework (Hsieh & Shannon, 2005, p. 1281). Summative content analysis derives initial coding from the counting of identified keywords which is not applicable if the interviewees use IT buzz words.

3.3 Case Study Planning

3.3.1 Case Study Design Setup

The lack of real-life reuse cases in literature (Brown et al., 2021; Keswani et al., 2014) has been partially addressed: Spoelstra (2011) presented one case at a medium-sized software enterprise by interviewing 6 experts. Bauer (2016) investigated reuse in two enterprises via interviews and questionnaires on various hierarchy levels. Both studied a general reuse experience that can include feedback from a long time ago or hearsay.

To study the reuse experience of experts recently hands-on involved in a specific reuse situation, this research has been set up in a multi-case design as an embedded case study (Yin, 2018, Figure 2.4, Type 4). Multiple units of analysis allow literal replication, similar results across cases, as well as theoretical replication to be studied. These are contrasting results that can be explained and give the study depth and compelling evidence.

There are two main reuse benefits: revenue or savings. Sales, solutioning, development and delivery are depicted in Figure 3.2, representing potential reuse initiation situations. While reuse can happen during all of these phases (Wolfram et al., 2020), only the first two phases, sales and solutioning, allow desired revenue to be generated by licensing the software asset and charging for the services. Then the contract is closed. The matrix of "Business benefits" versus "Reuse initiation situation" spans a field of six cases. However, once a deal has been closed, software asset reuse can only generate company-internal savings, diminishing the number of cases to five (see field "n/a" in Table 3.3).



Figure 3.2 CRM process stages - adapted from (Salesmate, 2018)

The unit of analysis is a reuse case. Each software asset represents one of the five real-life individual reuse cases in Table 3.3, thus shaping the holistic case study research (Yin, 2018). All cases are obtained within one case company, a global IT company.

Table 3.3 Types of reuse cases to be selected for this case study (source: author)

Reuse initiated during... Business benefit ...	Sales	Solutioning	Development/Delivery (after deal closure)
Generating revenue when reused (and some savings)	Asset Reuse Case 2 4 Interviews: C2-I1 Asset Creation C2-I2 Asset Storage C2-I3 Asset Distribution C2-I4 Asset Reuse	Global IT Company Asset Reuse Case 3 4 Interviews: C3-I1 Asset Creation C3-I2 Asset Storage C3-I3 Asset Distribution C3-I4 Asset Reuse	n/a
Generating savings when reused (and no revenue)	Pilot/ Reuse Case 1 4 Interviews: C1-I1 Asset Creation C1-I2 Asset Storage C1-I3 Asset Distribution C1-I4 Asset Reuse	Asset Reuse Case 4 4 Interviews: C4-I1 Asset Creation C4-I2 Asset Storage C4-I3 Asset Distribution C4-I4 Asset Reuse	Asset Reuse Case 5 4 Interviews: C5-I1 Asset Creation C5-I2 Asset Storage C5-I3 Asset Distribution C5-I4 Asset Reuse

To ensure rich research input, each case was captured via four individual one-hour interviews following the software reuse processes detailed in Section 2.5: asset creation, asset storage, asset distribution and asset reuse, which are based on the knowledge

management process exercise described in Section 2.4.4. Hence a maximum of up to four different individuals and a minimum of one could have been interviewed per each case.

The case study outcome does not provide “clear cut-solutions” as the “objective of case studies is learning through discussion, exploration and the search for intelligent questions” (Rasche & Seisreiner, 2018, p. 9). Therefore, the outcome are management guidelines listing reasonable approaches allowing teams to head in the right direction while narrowing their search for systematic reuse. The knowledge case studies provide, given the limited area of expertise, goes beyond statistical significance. A proper setup of the case study allows deep insights, specific conclusions to be drawn from the findings, as well as minimal bias by the researcher (Runeson & Höst, 2009).

3.3.2 Case Study Data Collection Procedure

The outcome of case study research depends on its upfront planning and clear setup of the semi-structured interview (Runeson & Höst, 2009). The semi-structured interview was prepared following Kallio’s (2016) guide which comprises five phases. First, the researcher identified the prerequisites that must be met for conducting a semi-structured interview. Second, existing knowledge in this area of research had to be retrieved and applied. Third, the author generated a preliminary – pilot – semi-structured interview guide for data collection. Fourth, the semi-structured interview was piloted and updated following the pilot findings for Case 1 in Section 4.2. “Appendix B: Semi-structured Interview Guide” represents the final *semi-structured interview guide* listing all interview questions.

In this qualitative case study research, *case selection* was done intentionally (Runeson & Höst, 2009), not guided statistically (Eisenhardt, 1989) but by case objectivity as well as for theoretical and paradigmatic reasons (Flyvbjerg, 2011; Redman-Maclaren et al., 2014). Flyvbjerg (2011, p. 397) sees “paradigmatic cases and case studies as central to human learning” as they represent a typical example. They allow for a comparative case study (Runeson & Höst, 2009). As a pragmatist, the author wants to “present [each] case for its value in the creation of knowledge” (Kaushik & Walsh, 2019) on software asset reuse. She believes that the gained knowledge is not ubiquitous. Instead, it forms just another step towards the universal truth about software asset reuse. By applying “abductive reasoning that moves back and forth between deduction and induction” (Kaushik & Walsh, 2019, p. 6), the researcher gradually builds up knowledge.

The selected cases represent five recent, real-world, business-relevant, and commercially successful reuse cases. Each case is based on a different software asset – diverse in terms of roll-out model, geographical and industry coverage, team size, age or complexity. To the knowledge of the author, this is the first in-depth presentation and analysis of this kind in the literature.

It would not have helped to investigate situations where the asset was not suitable because these cases offer less to learn from a business process perspective. They failed for a reason that is not within the scope of this study. These cases are almost never documented and hence difficult to track and trace.

The *interviewee selection* is determined by the case. In bigger asset teams there is a choice. Here the selection depends on factors like experience, availability or the interviewee's interest in contributing to research. In parallel, the researcher strives for interviewee diversity in terms of location, culture, ethnicity, age and gender. In contrast to many earlier pieces of reuse research, all the interviewees are practitioners.

Data collection via interview can be done through a face-to-face or online interview. Both types have specific advantages and limitations that must be considered.

The online interview offers cost and time efficiencies – no travel is required. It gives more flexibility in space and time – it can easily be postponed, interrupted and continued later. During web-conferencing, the participants can remain in their own premises, making the virtual interview very safe from a health point of view due to the ongoing Covid-19 pandemic (Lobe et al., 2020). The virtual interview gives both the interviewee and the interviewer the additional ability to easily screen share, look up facts online to answer or match things, and take notes without confusing the interviewee (Archibald et al., 2019; Deakin & Wakefield, 2014; Stephens, 2012; Weller, 2017). Finally, a web conference can be easily recorded and transcribed for later analysis. However, downsides can include the lack of non-verbal cues, the problem of easily relating to participants, and transient connectivity or audio issues.

In contrast, face-to-face interviews offer the benefit of creating a closer connection between participants and observing non-verbal cues live. This can be an advantage to get more detailed information and build trust. Still, the potential bias from the researcher's presence as well as travel to distant locations can be disadvantageous.

After weighing all the pros and cons, a *virtual data collection* method was chosen instead of personal data collection. The interviews were conducted online using a web conferencing system (Archibald et al., 2019) that the featured IT company often uses for business purposes.

3.3.3 Case Study Quality Assurance

The research setup and design determine the trustworthiness of the research results in the light of realistic research limitations (Runeson & Höst, 2009). Criteria have been introduced to perform a research design test. Yin (2018) suggests four design tests to evaluate a case study design: construct validity, internal validity, external validity and reliability.

Construct validity describes how well a topic is researched versus what is supposed to be researched. Operational measures must be taken to prevent subjectivity in data collection so that findings are generally valid in related cases.

Yin (2018) recommends taking two steps to construct validity, truth, in qualitative research. One step is what Flyvbjerg (2011, p. 396) calls “critical case selection”. Applied to this study, this means defining the software asset reuse in each selected case as a specific concept following the logic of Section 2.5 and explaining how it relates to the study objective of understanding critical knowledge management processes of software asset reuse in a global IT enterprise. All cases have in common that the software asset has been reused many times before. This means, the interviewees based their interview answers on a rich reuse experience. This rich and overlapping input allows for pattern matching and rival explanations in Chapter 4.

Another step is to determine what operational measures fit the concept. Prior studies have looked into the reuse of software assets (Bauer & Vetro, (2016); Svahnberg & Gorschek, (2017)). These, along with other relevant studies in the literature, have been used to comment on and validate the findings in Chapter 4.

False memory can be an issue in interviews (McGuire, 2022). This can be compensated for by the multi-case design, i.e., by comparing the findings from asking the same questions to multiple people within one case and across all cases.

Three tactics (Yin, 2018) reduce the danger of generalising from case studies (Wikfeldt, 1993). The first is multiple sources of evidence. Subsequently, the author interviewed experts with diverse backgrounds in terms of role, location, gender and subject matter expertise. Second, keeping track of the chain of evidence allows forensic insights into how the case study findings were derived. These pieces of evidence provide the logical connection, both back and forth, between the case study findings in Section 4.7, the case study database in NVivo as shown in Figure 4.1 and the WebEx transcripts used as citations in Sections 4.2–4.6. Third, having experts review the case study outcome to ensure that the data collected during the semi-structured interviews is unbiased, given the researcher is an employee of the IT organisation providing the case data. Detailed research results were presented to interviewees C2-I1 and C3-I1 for review and to more than 200 reuse experts during a subsequent one-hour web-session. The feedback gained ensures multi-perspectivity by considering different interests, skills and perspectives (Vasiljuk & Budke, 2021; Vogl et al., 2018).

Triangulation, “taking different angles towards the studied object and thus providing a broader picture” (Runeson & Höst, 2009, p. 136), increases research precision and construct validity. It counteracts the fact that primary case study input is qualitative data which is more comprehensive and richer but less precise than quantitative data. Additionally, this case study uses data source triangulation as the input data was collected in separate interviews with unique and diverse individuals from different locations who contributed specific work experience in software asset reuse.

The author of this study always acted as the interviewer. This prevents observer triangulation and adds quality. Further, the author compiled the interview questions in the semi-structured interview guide using different perspectives, e.g., the view of the asset reuser, the global company and the asset owning team, to allow for viewpoint triangulation.

Internal validity refers to research consistency. It gives confidence in the truthfulness of research findings (Noble & Smith, 2015) and is achieved when experts not involved in the research can recognise and conclude the phenomenon being studied based on the descriptions and interpretations provided. Section 3.3 describes the qualitative case study setup in a way other researchers can understand and replicate.

Yin (2018) lists four tactical data analysis methods to achieve internal validity in case studies: pattern matching, rival explanations, explanation building and logic models. These are described as data analysis methods in Section 3.2.5 and applied in Section 4.7 when case study findings are matched against each other and against literature findings. Explanation building and logic models are provided in more detail in the answers to the research questions in Section 4.8 when tables or figures are shared.

Consistency of the presented evidence is demonstrated across cause and effect when data analysis employs tactics such as pattern matching, rival explanation checking, explanation building and logic models (Yin, 2018), as described in Section 3.2.5 and implemented in Section 4.7. By adding flow charts and figures, the researcher provides supplemental explanations and logic models, thus offering a different perspective on the research findings.

External validity is addressed during the research design and describes the applicability or generalisability of the research findings to a context outside the investigated cases (Noble & Smith, 2015). External validity is achieved by mapping research findings against theory in Sections 4.2–4.7 and implementing the concept of a multiple-case study where cases are compared against each other as described in Section 3.3.1.

This multi-case study synthesises evidence of five individual asset reuse cases collected in the context of interviews with 19 individuals who are experts in different roles and levels for the selected IT provider. Some of the findings show up in all five cases. Therefore, they can at least partly extend to similar cases in comparable contexts.

Interviewing employees of the same IT provider threatens external validity. There is a danger of closed-mindedness due to groupthink. However, each of the 19 experts was interviewed individually, thus minimising groupthink (Akhmad et al., 2021) during interviewing. Further, the asset teams do not know each other. They all come from different parts of the organisation and projects.

Yin (2018) suggests *reliability* as a fourth case study design test. Noble & Smith (2015) call this consistency. In a qualitative study, reliability describes the dependability of the study methods, the credibility of the findings and the consistency of data interpretation.

Qualitative research is less precise and quantifiable than quantitative research which impacts reliability. The design test uses similar tactics like construct validity – ensuring

that given a similar set of data, other researchers would repeatedly come up with akin results and conclusions as listed in Sections 4.7, 4.8 and Chapter 5. Subjective bias is minimised by providing direct interview quotes. The case study setup detailed in Section 3.3.1 allows other researchers to follow up and achieve similar research results.

Researching the IT organisation, the author is employed at, gave access to reuse cases and respective expert interviewees with a detail that could not have been gained otherwise. Since this IT company is staffed with IT experts who have worked at previous employers, the research reflects an international, multi-corporation standard. The identified findings align with the literature covering reuse at competitive IT providers.

Consistency of data interpretation is based on neutrality. Neutrality captures the likelihood of reproducing the results achieved in this case study under similar circumstances, excluding bias and other influential circumstances. It is sometimes also referred to as the confirmability (Noble & Smith, 2015) and has to be addressed early, during data collection, applying similar measures as for truth value (Yin, 2018). A multi-case study setup typically helps to demonstrate a decision trail, thus providing trust in this case study research regarding the replication logic of the cases (Noble & Smith, 2015).

3.3.4 Pilot Study

The purpose of the pilot study is to test and conceptually clarify the research design and suggest improvements in quality and efficiency. A pilot study verifies the theoretical framework, the overall research approach, with minimum effort. Further, it checks the crafted semi-structured interview guide both from a user and timing perspective, to improve it in depth and scope and test it with the planned method of analysis. To benefit from its effects, the pilot study took place well in advance of the entire case study (In, 2017; McIntosh & Morse, 2015; Yin, 2018).

According to Kallio et al. (2016) piloting could be done either internally, by experts, or in the field. Internal testing was done upfront when the author of this study assumed the role of an interviewee to test the complexity and the flow of the set of interview questions. A more powerful field-testing took place when a pilot study on one reuse case, Case 1 in 4.2, was conducted.

While the pilot case study delivered results which can be used for the final case study, changes had been identified to the initially planned research methodology. These changes applied mainly to *data collection*, the *setup* and conducting of the *interviews*.

Because the asset team in Case 1 comprises only a small team, the researcher interviewed one person twice – once on asset creation and then in her role as marketing lead on asset distribution. While she was an expert on both topics, this may have limited the scope of the interviews. Not surprisingly, the interviewee expressed the same views in both her interviews when she was asked related questions. Interviewing different experts adds breadth and depth to the case capturing and allows more diverse views to be gained. This was revised during the full case study – all other cases were conducted by interviewing four individuals.

Improvements were introduced to the interview setup as a result of piloting. For example, a few questions in the “Appendix B: Semi-structured Interview Guide” were changed from closed format to open questions.

One pilot interviewee (C1-I2) could not answer all the questions. However, a closer look revealed that the questions asked were more related to asset distribution and were therefore permanently moved to a different section of the interview guide.

Conducting the interview, the researcher was initially nervous about covering all the questions within the agreed 60-minute time frame. This was stressful for both the interviewee and the interviewer. Every new question interrupted the interview flow. It turned out letting the interviewee talk for some while generated quantitatively more and qualitatively better interview content. Additionally, it increased the satisfaction and well-being of the interviewees. Some of them wished to carry on the interview although the final question had been asked.

3.4 Ethics

Prior to each interview, the researcher shared a three-page document with each interviewee containing the ethics letter with information on the intent of the research, the research sponsorship letter from the global asset organisation of the IT company as well as the consent form to be approved by each interviewee. It was made clear that participation was voluntary. Since these interviews did not focus on personal issues but on business topics, the main ethical consideration touched on how confidential

information would be dealt with (Runeson & Höst, 2009). The letter of consent follows the guidelines issued by Heriot-Watt University. Further, the researcher explained at the start of each interview that the names of all people, assets, clients and competitors would be removed prior to publication so that no interviewee or asset could be identified.

The entire set of interview questions was shared upfront with each interviewee (Lobe et al., 2020; McIntosh & Morse, 2015). The questions asked have a minimal vulnerability potential. Although the interviewees' answers reflect personal opinions, they are business oriented (Roth & von Unger, 2018).

Amschler Andrews & Pradhan (2001) describe ethical issues inherent in software engineering research that also apply to research in software management. These issues mainly centre around how insights into software can be misused by the researcher. Further, the IT company's restrictions could prevent the researcher from disclosing the software-related insights that are required to understand the piece of research or a delay in publication which diminishes the impact of the research. While there can be guidelines and policies in place, the authors point out that "credible results ... are based on mutual trust that everyone will behave ethically" (Amschler Andrews & Pradhan, 2001, p. 109).

4. Case Study Presentation, Findings and Discussion

By discussing knowledge-related processes of real-life software asset reuses from a business point of view, Chapter 4 addresses all three identified research gaps. Section 4.1 provides application-related content of the case study execution and analysis. Sections 4.2 - 4.6 detail the findings of the five cases that constitute the case study. Section 4.7 discusses the findings by contrasting all cases. Finally, Section 4.8 supplies the answers to the research questions.

4.1 Case Study Presentation

4.1.1 Case Study Execution

Conducting the case study requires selecting the right case company, cases, and interviewees. *Choosing the right case company* is essential to ensure the relevance of the research and its quality. Several selection criteria play a role such as, e.g., the alignment with the research topic, representation of the research industry, willingness to participate and interest in this research, access to detailed research data, diverse perspectives, feasibility and the interviewer's background (Aithal, 2017; Rashid et al., 2019).

Alignment with the research topic means that studying the case company will contribute to knowledge by providing useful insights for this research.

The selected case company should represent the right industry so that phenomena relevant to the targeted research can be studied. IT reuse maturity plays a role. Therefore, studying an IT vendor operating an established software reuse business is beneficial. In this way, insights can be gained about those knowledge-related reuse processes that form the basis for the success of reuse.

Ethics matter in choosing the case company. This starts with the general willingness of the company to support this research, their interest in the research topic, and ensuring confidentiality in the business details of this company. The case company should cooperate with this research and be willing to share information.

The case company has to ensure adequate access to reliable and relevant information for conducting the case study. General case accessibility is important. Related factors are resource availability, research enthusiasm, and company management support. Further, specific data access matters. A sufficient and relevant record of successfully reused

software assets is required to select cases. Insightful selection data is a prerequisite for research success, as it allows for locating practical actors in software asset reuse rather than capturing meta-information from contacts with hearsay insights.

The company chosen for this case study should offer diverse perspectives on the studied problem to allow for rich analysis and different viewpoints for comparison and triangulation. The size of the company matters. Large enough companies have a broader reuse potential and often more established software reuse practices.

Time constraints, logistics, and other feasibility factors in conducting the case study play a role when choosing the right case company. The case company should have a state-of-the-art software reuse practice that can be studied remotely. The aim is to study recently completed reuse cases within a favourable research period.

Further, the researcher's background should match the case company. It helps to have company insight. Understanding the complex business of the case company, being able to speak the corporate language, understanding its many acronyms and being familiar with its internal workings helps when conducting interviews and during interview analysis.

Using these criteria ensures that the data obtained is sufficient to support this software asset reuse research and that the company selected for this research is appropriate. Other factors, such as the nature of the client industry, did not influence the selection of the IT provider to be researched.

Applying the above company case selection criteria and following the “learn from the best”-credo (Bernard et al., 2014), this case study was conducted as a one-case company study at one of the leading IT providers known for reusing software assets for decades. It is a global enterprise offering innovative IT services in over 170 countries. The company's profile is diverse, putting a current focus on business consulting, cloud computing and artificial intelligence.

Selecting the right cases is important for creating a complete case set for the intended embedded multi-case case study. The research quality depends on the case selection criteria applied. One criterion is the geographical reuse scenario. Reuses that span a geographical distance typically require more systematic reuse planning to achieve cross-

geo knowledge distribution. Hence these cases allow studying a higher level of systematic reuse.

Further, asset commercialisation plays a role. Fully commercialised assets are of greater quality having undergone procedures like mapping against an offering, pricing, name knock-out search and others. These assets are entered in the company-wide asset repository, which provides marketing and educational material on the asset. These cases enable to better study the planning of systematic software asset reuse.

More, each presented reuse should be a new reuse case, i.e., not a contract extension. This is important for analysing and interpreting the asset management processes.

Applying these case selection criteria onto cases fitting the characteristics listed in Section 3.3.1 increases the research quality. This is how the software asset reuse cases 1 – 5 got selected out of ca. 100 generally relevant cases while the researcher checked reuse documentation and talked to knowledgeable employees of the IT provider.

Case identification drives the *selection of the interviewees*. Per each reuse case, four distinct experts had to be identified – for software asset creation, storage, distribution, and reuse. The choice is limited if asset teams are small and there are not many experts to choose from, e.g., Case 1 and 5.

Expertise is another selection criteria. An interviewee should be an expert in his area. Long team affiliation drives expertise. As a side effect it increases the number of witnessed reuses which allows interviewees to comment on the reuse case more accentuated.

Finally, individual aptitude plays a role. This includes factors like the use of English language, job motivation as well as the desire to contribute to research. Highly motivated experts who can explain their reuse case in good English grant a richer interview yield.

Applying these factors to the interviewee selection enhances the research quality. 29 experts were contacted – 10 to verify the suitability of cases out of which 5 were refused because the case was skipped. Out of the remaining 24 experts, 19 (see Table 4.1) were selected applying the selection criteria listed in Section 3.3.2. These 19 interviewees participated in the actual case interviews listed in Table 4.2. Table 4.1 details their role, location and team before they will be more closely featured during the case presentations



in Sections 4.2 - 4.6. To allow an easy case comparison, Table 4.2. summarises the asset details in terms of reuse starting year, location of creation and current team size, as well as the case details in terms of industry, reuse country and reuse team. Further, it maps the interviewees introduced in Table 4.1 against the cases. For this research, within the given set of reuse experts, their individual background, education, age, nationality, number of patents or similar had no impact on their interview qualities.

Table 4.1 Interviewee list (source: author)

Interviewee	Gender	Role	Based in	Team
C1-I1	f	Long-term asset owner witnessing asset creation	US	Asset Owning Team
C1-I2	m	Asset developer from the start	US	Asset Owning Team
C1-I3 = C1-I1	f	Asset marketing	US	Asset Owning Team
C1-I4	f	Corporate audit manager reusing the asset	China	Client
C2-I1	m	Asset brainstormer, founder, technology leader	US	Asset Owning Team
C2-I2	m	Asset lead developer, managing a team of developers	Czech Republic	Asset Owning Team
C2-I3	f	Global marketing and engagement lead for the asset	Netherlands	Asset Owning Team
C2-I4	m	Banking expert selling the asset to the client	Belgium	Account Team
C3-I1	m	Quality Engineering Sales lead, asset brainstormer and founder	Australia	Asset Owning Team
C3-I2	f	Asset offering and development lead	India	Asset Owning Team
C3-I3	f	Testing process lead	India	Asset Owning Team
C3-I4	m	Business Sales & Delivery Executive suggesting the asset to the client	Czech Republic	Account Team
C4-I1	m	Program Manager 3rd Party Integration	Ireland	Asset Owning Team
C4-I2	m	Technical architect	India	Asset Owning Team
C4-I3	m	Complex deal leader and project launch leader	Ireland	Asset Owning Team
C4-I4	f	3rd Party Software consultant reusing the asset	Sweden	Account Team
C5-I1	f	Technical Program Leader and asset creator	India	Asset Owning Team
C5-I2	m	Developer finance applications; test automation	India	Asset Owning Team
C5-I3	f	Test lead promoting the asset	India	Asset Owning Team
C5-I4	m	Delivery project manager suggesting and reusing the asset	India	Account Team

Each interview kicked off with some *small talk*, asking for voluntary participation and establishing rapport to confirm the relationship (Archibald et al., 2019; McGrath et al., 2019). This was followed by a short introduction to the research and details of the consent form (McGrath et al., 2019). This part of the session was not recorded.

Table 4.2 Overview on the five cases constituting this case study (source: author; numbers provided for 2021)

Reuse initiation situation Company-owned asset ...	Seller initiated reuse	Solutioning Team initiated reuse	Developer/delivery initiated reuse
Generating revenue when reused  and some internal savings	Asset 2 for Case 2 <u>Asset 2</u> Initially from 2015 Commercialised Created in: US + global (US + Europe) Asset Team: ca. 100 experts <u>Case 2</u> Industry: Banking Reused in: Belgium Reuse team: external Banking client Interviewees C2-I1 C2-I2 C2-I3 C2-I4	Asset 3 for Case 3 <u>Asset 3</u> Initially from 2015 Commercialised Created in: Global (US, Australia, India) Asset Team: ca. 35 - 39 experts <u>Case 3</u> Industry: Banking Reused in: Estonia Reuse team: ca. 50 experts reuse the asset Interviewees C3-I1 C3-I2 C3-I3 C3-I4	n/a
... generating internal savings 	Asset 1 for Case 1 - Pilot Case <u>Asset 1</u> Initially from 1992 Commercialised Created in: US Asset Team: 6 experts <u>Case 1</u> Industry: IT (internal) Reused in: China Reuse team: Chinese IT team of the IT provider Interviewees C1-I1 C1-I2 C1-I3: same as CI-I1 C1-I4	Asset 4 for Case 4 <u>Asset 4</u> Initially from ca. 2012 Commercialised Created in: US Asset team size: 126 experts <u>Case 4</u> Industry: Industrial Reused in: Sweden Reuse team ca. 90 people out of which ca. 40 – 50 are functional consultants Interviewees C4-I1 C4-I2 C4-I3 C4-I4	Asset 5 for Case 5 <u>Asset 5</u> Initially from 2016 Commercialised Created in: India Asset Team: ca. 4 experts <u>Case 5</u> Industry: Travel & Transportation Reused in: Canada Interviewees C5-I1 C5-I2 C5-I3 C5-I4

All interviews were conducted as *web conferencing* sessions – recording sound, video, and voice to text transcripts and storing these in a way that is accessible only to the

researcher. The transcript file was quality checked and edited during transcript validation. All interviews minimised the personal conversation typically used to establish a relationship (Goffman, 1981), to make the most of the one hour scheduled for the interview. Several interviewees switched on their camera, which helped with social bonding and improved the communication outcome due to the “McGurk effect” that speech can also be seen (Barr, 2013). On the other hand, switching cameras on does not fully replace the missing eye-to-eye contact due to the gap between the camera and the video picture of the conversation partner (Vrzakova et al., 2021).

Each *citation* is a revised transcript that was anonymised by pseudonymising the names of mentioned people, companies, or competitors (Roth & von Unger, 2018). All interviewees were given code names in the form of “Case number – Interview number” to pseudo-identify them (Roth & von Unger, 2018).

4.1.2 Case Study Analysis

Analysing representative reuse cases and comparing them against each other can allow relevant *reuse patterns* to be recognised. The subsequently identified reuse processes allow conclusions to be drawn on what systematic software asset reuse looks like in practice and what processes would be required.

The researcher analysed the *transcripts* of the recorded semi-structured interviews using thematic analysis and the NVivo software package. First, the author checked the automatically derived transcripts for completeness by running each interview replay at half the recording speed while editing the transcribed text to reflect as closely as possible what each interviewee said. This check allowed a deep dive into the content.

Second, the researcher coded all transcripts in NVivo. Based on this, initial codes were identified. These can be seen in the blue framed column in Figure 4.1 – marked by NVivo with a blue ball.

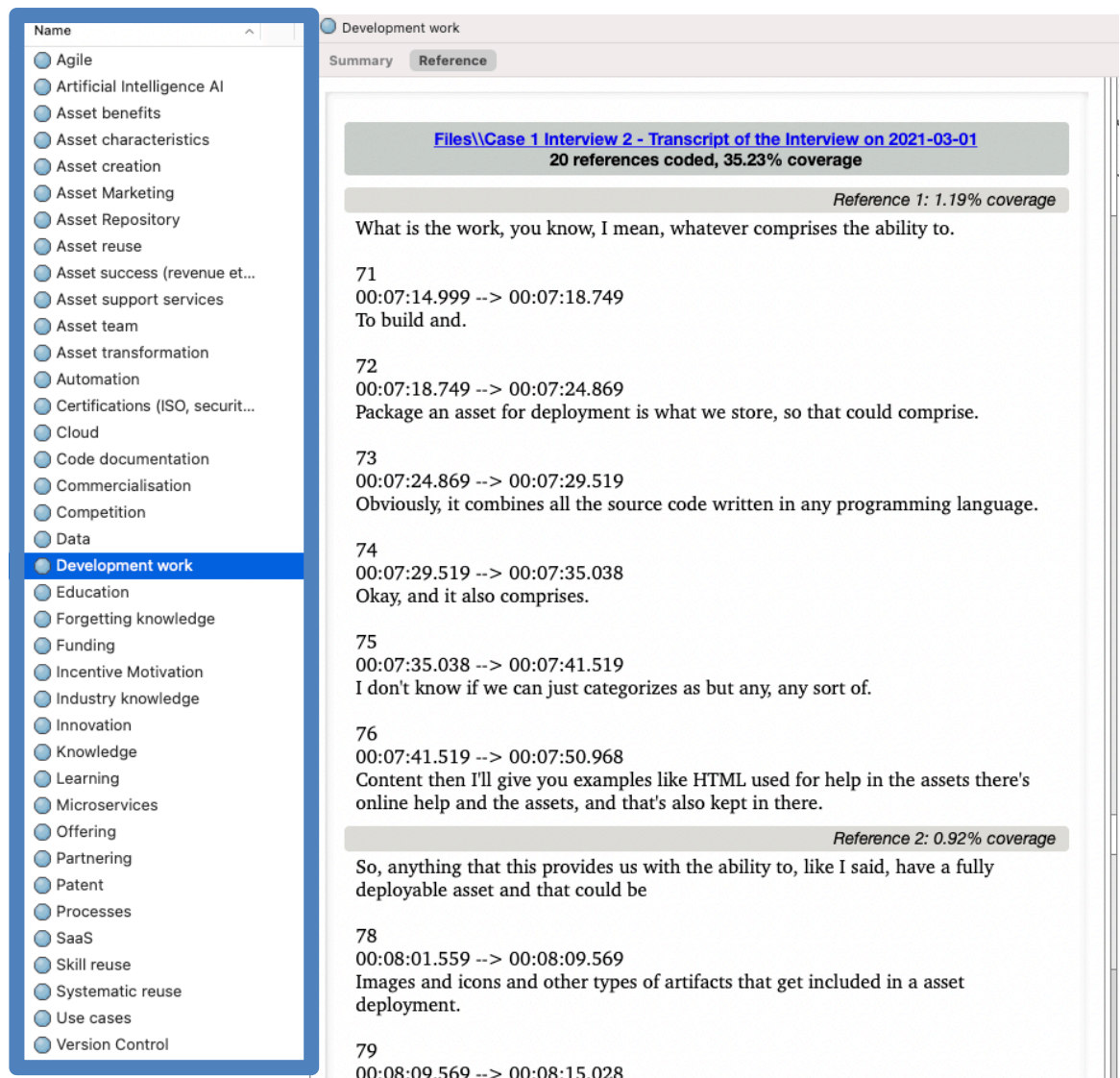


Figure 4.1 Screenshot of NVivo analysis – identified codes appear in the left column (source: author)

Third, after all data had been initially coded, topics, Braun & Clarke (2006) call themes, were identified. They represent patterns across the data set. Identifying topics was done via interpretive analysis, i.e., the researcher sought to understand the content, underlying message and significance of the respective interview text section. For this analysis, the author applied reflection, critical thinking, and business knowledge to cope with the granularity and complexity of the research subject. The analysis itself was done via an Excel table, as shown in Figure 4.2. Here the researcher went in intensive work back and forth between the coded interviews and the pattern search while identifying main topics (marked in light blue in Figure 4.2) and sub-topics (listed in white in Figure 4.2).

Fourth, while populating the table in Figure 4.2 to prepare the case presentation for Sections 4.2 - 4.6 and the case comparison for Section 4.7, the researcher reviewed the topics to get a better understanding of the nuances of each interview. The coding was checked and adapted to reflect all relevant research input.

Fifth, the identified topics were refined and confirmed. Consequently, 38 codes in NVivo transformed into 19 main topics, out of which the screenshot in Figure 4.2 shows only 14, marked in light blue on the left column in Figure 4.2. These main topics and their sub-topics were mapped against the four major asset reuse phases: creation, storage, distribution and application (dark blue bars in Figure 4.2, the screenshot shows only three major reuse phases), and reported in the analysis section of this thesis (Braun & Clarke, 2006).

Sixth, Section 4.7 compares the findings across the five cases. This allows the reasons for the differences (Eilbert & Lafronza, 2005) to be explained. Both similar subprocesses (literal replication) as well as contrasting subprocesses (theoretical replication) can be analysed (Yin, 2018).

Following an abductive research approach, this case study setup allows theories to be formed, by looking at a causal sequence of the reuse subprocesses, by time-series analysis on the observed versus theoretical series of reuse process events or by analysing logic models by investigating causal relationships between reuse subprocesses (Eisenhardt, 1989; Yin, 2018).

Case	1	C2	C3	C4	C5
Asset Creation					
The asset itself					
Initial objective	help research to help a client	Create from scratch a soft	To stop the never ending	Replace an earlier asset tha	Stop flood of small as
Check internally/externally	- there was nothing	y - there were many other	y - there were 6-7 smaller	y - there was in the nineties	y - there are many, m
Who came up with the idea	team of 2-3 experts (incl. res	Lead developer + executive	Thought leaders (3 execu	In the US - there were 3 ma	a handful of testing e
Competition	- both internally (partnering	y - no internal competitio	y - constantly new tools	y - there was an old asset w	y - there are many, m
Asset Team					
Asset team size	full-time: 4 - 6? people	full-time: 60 developers	ca. 35 39; not full time: 1	126	part time/other roles
Team = ...	team = family/centre of com	Team = like a start-up	Huge organisation - Regi	it's a big team ... some peo	Team = like a family
Team stability	y	y	y ... but it's a big team (k	it's a big team ... some peo	y
Motivation/Awards	business success + totally nev	1. help the client 2. freed	1. high visibility due to so	asset team received some r	1. management atten
		- perceived awards as a	- perceived awards as a	good marketing tool	
Learning	Academy learning	Bootcamp for newcomers	AI and machine learning	Lessons learnt from previ	struggle to learn the s
Number of reusers	Asset Owners	Asset Owners	Asset Owners + Account	Asset Owners + Account tea	Account team only
Asset sponsor	see the global asset team as	2 sponsors - one from bus	has a sponsor	sponsor is global service lin	Global Test Automati
Funding					
Funding	externally by the client or inte	combination of clients (in	incremental funding	corporation spent money on	receive claim code fro
Asset Coding and Storage					
Development work/Storage					
Asset changes made	only by asset owning team	only by asset owning team	only by asset owning team	only by asset owning team	in the repository only
Code enhancements	internally imposed: continuous	internally-imposed: strong	incremental coding and h	internally imposed focussing	internally imposed: or
	asset enhancements by asset	asset enhancements by a	10th version now - enhance	ments by team and picking	priority on generic fea
	asset creators/developers = h	customisation, microservi	development follows a dev plan		developers = technical
	y sharing new asset feature:	by sharing new asset feat	by sharing new asset feat	by sharing new asset feat	by sharing new asset
Partnering		y - for cost reasons: part	y - very active		y - e.g. with Oracle; w
Code Documentation	code library system for versio	follow coding standards	y in the form of raw files	No coding language used; E	no traditional code - it
Knowledge					
Knowledge	content knowledge/industry	Content knowledge	methodology knowledge,	methodology knowledge	methodology knowled
	reuse gives access to asset AND	team	Issued a badge on the asset	knowledge that internal	team participated in f
Asset Repository					
Forgetting knowledge					
Forgetting knowledge	no forgetting - original asset	No forgetting but full cod	Knowledge is preserved in	No forgetting - all initial doc	No forgetting of know
	knowledge = a building		Some accounts developed	knowledge the asset team	decided to skip and not
How the asset keeps					
different code for each		n - one cloud-set-up for a	y - will be client-customis	n - but it is customised	
Transformation					
Transformation	horizontal reuse = repurpose	y - very flexible			
Brother/sister assets	- (1) because it is specific	n - not industry specific	n - asset had been kept a	y - for various industries	y - for mainframe, mc
Technology	externally imposed: Code lang	Externally imposed: Code	language changed/code r	Externally imposed: every qu	Externally imposed: di
Innovation					
Innovation	innovation = natural progress	comes from the (1) asset	comes from the (1) asset	comes from the (1) asset ov	comes from the (1) as
Automation					
Automation	to come	y - e.g. for fast deployanc	y - the asset benefits from	y - automated testing; auto	y - test automation; i
AI	Asset Repositories		y	n - no use of AI; the 3rd part	n - no use of AI
Agile		y - agile work mode	y	y - agile work mode incl. lau	y
Cloud					
Cloud	- asset is available on on pri	y - asset is on cloud	y - asset is on cloud	y - asset on cloud	soon to be on cloud; n
	each client has its own cloud	one cloud set up for all (multi-tenant set-up) + customisation			
	have to break with the rule and	give the DevOps team access to the asset			
Microservices		Microservices	y		
Security & Certifications	Security and Certifications	Security and Certifications	(ISO 27001) took 1 year	- important when external cl	Security effort vs oper
Patent					
Patent	- perceived as not importan	y - a few exist; team is a	n - Wish they would have	n - no patents	n - no patents
Asset Distribution					
General Marketing					
Use Cases			helpful during client presentations to illustrate the value-add of the asset w		
References		y - one really good reference		y	References/reuse suc
Award as marketing tool	see (see line 26)				
Asset Organisation	- is a great support				
Dedicated marketing budget		N - no dedicated marketing budget			
Selecting a suitable asset	- rather not since it points to an industry in ASP		? - rather not		
Internal Marketing by the asset owning team					
Rate asset team find	0:50			50-50	60% asset team locat
Awareness sessions:	- internal sessions + commu	Community Calls to inform	Y - everybody in the testir	Quarterly internal enableme	Regular connect-sessi
Training sessions	- asset is only manipulated	n - asset is only manipula	Y - training programs; asset is used by a wider team c	y - provided also to th	

Figure 4.2 Screenshot of Excel analysis – identified codes appear in the left column (source: author)

An additional criterion for interpreting the findings are rival explanations (Henry et al., 2013; Posey et al., 2011; Wright, 1973). This was done by mapping the case study findings against “the most plausible rivals” found in the related literature (Yin, 2018, p. xxii). Identifying and addressing these rivals added strength and rigour to this research. While this case study setup may not allow for the most unusual cases, the chosen approach gives the basis for a robust analysis (Herriott & Firestone, 1983) due to the multiple cases.

The researcher produced one flowchart per case based on the interview input. The gained flowcharts facilitate a case study technique called pattern matching (Yin, 2018) by mapping the subprocesses of each reuse case against the four asset reuse phases explained in Sections 2.5.2–2.5.5. This analysis identified that real-life subprocesses are handled in practice sometimes differently to the description in the literature, or they are not mentioned at all in the literature.

Based on the findings, the author derived a recommended ideal process for systematic reuse leverageable in business practice from this case study – the management guide.

4.1.3 Research Diagram

Figure 4.3 provides the research diagram summarising the stages of the research application in a sequential and logic order. RQ3, an unanswered question in practice, forms the research input. It is backed up by RQ1 and RQ2 – questions of theoretical and practical relevance that are gaps in research.

The literature research identifies: three areas of research, four fundamental knowledge management processes as well as the theoretical framework. All of these provide structure to the subsequent case study research.

Following an identified design, 10 real-life cases presented by 10 experts were scrutinised. Out of these, 5 met the prerequisites. For the pilot case 3 experts were interviewed (one person gave two interviews). In total 19 experts gave 20 recorded and later analysed interviews. The case study along with the literature findings form the research output. This output allows to answer the set research questions and to contribute to knowledge and practice. Further, it details the theoretical framework by describing meticulously how intellectual capital (the asset) is turned into innovation which improves the organisational performance.

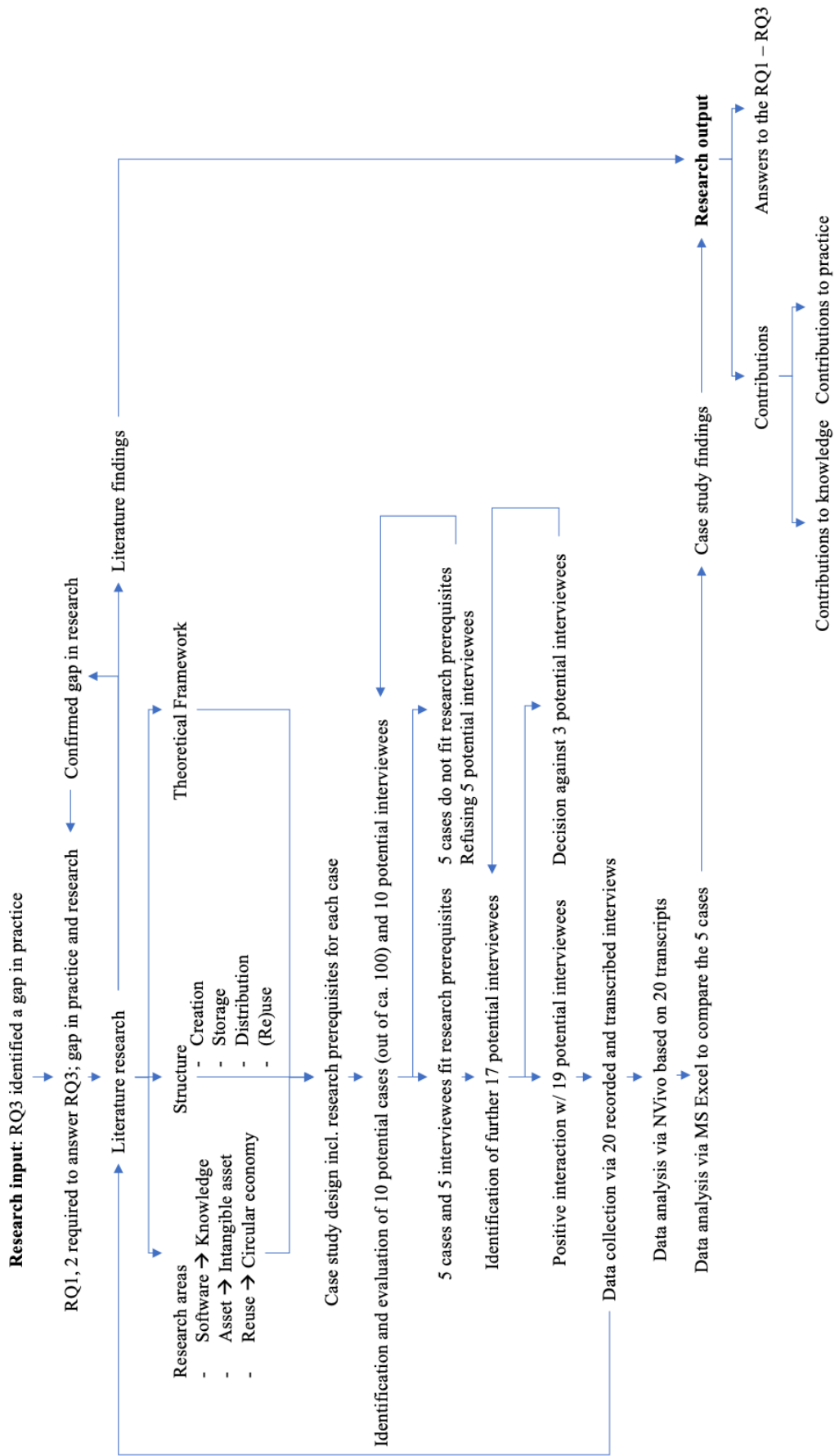


Figure 4.3 Summarised research approach (source: author)

4.2 Findings Case 1 – Reuse Initiated during Sales Generating Savings

Case 1 describes the real-life case of a small US-based team that created and still runs a software asset which was reused 29 years later in China (see Figure 4.4). This reuse was suggested during the sales cycle. Since the asset is reused company-internally, there are savings across all 320 locations of the IT provider in China, but no revenue. The company works within the IT industry, within the communication sector.



Figure 4.4 Case 1 - Location of asset owner versus asset reuser - adapted from (Geology.com, 2021)

The following interview data were gained through the pilot study but are here treated as part of the main data set.

4.2.1 Case 1 Asset Creation

Case 1 focusses on two main asset creation topics – the asset itself and the asset team (Figure 4.5). These topics will now be discussed in detail.

Discussion topics	Case 1
Asset Creation	
The asset itself	...
Asset team	...

Figure 4.5 Discussion topics of Section 4.2.1 (source: author)

The Asset Itself

Thirty years ago, the software asset was created as the first of its kind when hardware was sold to a client. At this time, the asset team supported research to develop the initial software asset (C1-I1). The team checked for existing assets prior to creating their asset. Since there was no such asset available, they developed the asset new for one specific client - see (1) in Figure 2.17 (Mateen et al., 2017). As mentioned in Section 2.4.5, *this level-setting step of checking for existing knowledge* has not been covered in the literature but is relevant and occurs in practice.

Creating the software required unique knowledge, took up to a year and a half and was done by two to three experts. This confirms Nonaka's (1994) theory mentioned in Section 2.4.3 that teams collaboratively create business knowledge based on the individual knowledge they possess. The client team was strongly involved in formulating the initial asset at this time. Consequently, the *software asset represents knowledge of the asset owning and of the client* organisation. This has not been covered in the literature, yet. Applying knowledge management (Wijnhoven, 2008), the IT provider contractually secured the IP rights.

Asset Team

In contrast to the transformative nature of the asset (see Section 4.2.2), the *asset team remained stable*. Interviewees shared that most people who joined the team left to retire and not before. This is in line with the researcher's work experience. However, it has not been covered in the literature. The fact is surprising since US-based employees tend to change their employer on average every 4.1 years (Statistics, 2020). The US-based asset team stayed not only for an unusually long time with the same employer but also in the team constellation. The interviewees C1-I1/2/3 described how their roles changed over time within the team, which relates to the dynamic capability theory (Marrucci et al., 2022). In line with what Zhou (2019) and Human (2020) write in general about teams, knowledge build-up and specialisation towards higher roles took place. The members of the asset owning team moved team-internally up after an established team member retired.

One asset owner attributed the success of the asset to the consistency in the team and the personal passion of each team member for the asset. She referred to the asset as "my

baby” (C1-I3, L474) and the team as “a family” (C1-I3, L445), caring for it. Epley (2007) explains that behind this anthropomorphism is *motivation* and passion but also limited connection to other teams.

The asset owning team feels very close and is extremely *motivated*. “Addressing the business case and generating revenue ... were the incentives.” (C1-I1). The reuser on the other side felt motivated as the reuse of the asset applies a “totally new concept” (C1-I4). Incentives did not act as motivators. This is in line with the literature (Fichman & Kemerer, 2001; Frakes & Fox, 1995; Kuo, 2013).

Securing team funding for the six asset team members out of the business success of the asset – either internally via a document of understanding granting charge codes or externally via contract volume – gives the team financial and general independence. This could potentially act as a strong motivator to stay together and carry on, indicating *autonomous job motivation* as explained in Section 2.4.7 (Bidee et al., 2013; Frakes & Fox, 1995; Gagne et al., 2019; Karim & Majid, 2018).

Learning plays an important role for the asset owning team. Asset developer C1-I2, for example, has included UdeMy learning in his daily routine for years. Bringing the software asset to the cloud requires substantial learning effort from the asset team. This learning effort aligns with Burciu & Kicsi’s (2015) findings. One asset team member stated, “... my management team ... is promoting the use of our skills – not necessarily the use of our assets” (C1-I3, L266-267). This confirms that the acquired knowledge turns the asset team as such into an asset, a centre of competence, as explained in Section 2.4.3 (Ram & Devi, 2019).

It is also worth mentioning the *skill reuse* triggered by asset reuse. The asset is owned by subject matter experts who combine a specific skill set that is quickly available and forms a competitive advantage in the market if needed.

4.2.2 Case 1 Asset Coding and Storage

Figure 4.6 presents the asset coding and storage topics the interviewees raised in Case 1. These will be discussed now.

Discussion topics	Case 1
Asset coding and storage	
Development work and storage	...
How the asset keeps on changing	...
Transformation	...
Innovation	...
Automation	...
Cloud	...

Figure 4.6 Discussion topics of Section 4.2.2 (source: author)

Development Work and Storage

Software development is a particular kind of *knowledge storage*. It turns implicit knowledge into code representing explicit knowledge, as described by Gamble (2020) and Majchrzak (2004) and explained in Section 2.4.3. Code can be stored, distributed, applied and reused.

Development ties the asset reusers and owners together during the years of reuse. The interviewed reuser C1-I4 reflected on her ongoing relationship with the asset owning team by stating that *asset enhancement* is an ever-ongoing process. Once Asset 1 was created in 1992 for the first client, the development set in and never stopped – leading to 400 customer-tailored versions, 40,000 features, and the emergence of related assets. This wide-ranging development work is not highlighted in the literature and goes beyond the simplification in Figure 2.16.

The continuous code changes need to be managed in terms of knowledge management (Wijnhoven, 2008). Two aspects relate to the dynamic capability theory (Marrucci et al., 2022): code version control management and the ability to deal with changing version control systems (C1-I2).

Using a repository in the form of a *code library system* is better than storing code locally (C1-I2). The interviewee underlined the need for good version control. Code versions grown over almost 30 years are still available and well under control.

Version control systems change from one system to another. Each change required work effort. Currently, the team is undertaking *version control management* on GitHub. While version control is relevant for all development work, the interviewed developer C1-I2 clarified that developing an asset is not just about creating new code but maintaining and

administering the code over decades - across library and system changes. This is a knowledge management aspect (Wijnhoven, 2008) that requires planning and investment not covered in the literature to date.

Reusable software assets offer an additional advantage compared to other forms of software. “When we develop something for a new client, a new implementation, we share that with our other clients as well. So, we push those models back to our other clients because we know it’ll be useful for them as well. ... And that’s consistent across our assets. ... We may or may not charge for it, but we have an annual maintenance agreement with our existing clients ..., and we agree that we will *share new updates and new capabilities available.*” (C1-I1, L270). So, the asset stays fresh and recent – clients don’t have to drop it in favour of alternatives. Reusing an asset can involve an innovative subscription model to receive updates from the provider and other peer clients. No literature could be found where this concept has been described to date.

The asset code is “pretty well documented” (C1-I1). In “... GitHub, each of the repositories has a read me file. That helps anyone that’s going in there to understand what the purpose of that repository is” (C1-I2). In contrast to earlier publications on software asset creation that identify the reuser as in charge of writing the extensions (Svahnberg & Gorschek, 2017), the developers keep the asset code and make all the changes themselves. This confirms Birch & Muniesa’s (2020) general asset management finding that asset creators maintain ownership and control throughout the asset life cycle.

When the asset emerged in 1992, there was no *asset meta-repository* yet. Today, it allows potential reusers to locate the asset and its owners. It contains no code, only asset metadata such as sales material or the asset licence fee (as AL Badareen (2021) described in Section 2.5.3).

How the Asset Keeps on Changing – Caused by the Asset

Resulting of the ongoing development, the asset has “gone through some *transformations* over the years” (C1-I1). This concerns multiple aspects. The asset’s initial code was not expanded in one but in various directions, as the asset covered more industries; this confirms the horizontal reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010). The circular economy literature refers to this as repurposing (Van Buren et al., 2016). This

asset evolution resulted from the ingenuity of the IT company's asset owners, their clients' feedback and the indirect collaboration and interaction of the client group.

Transformation – Externally Caused

During the almost three decades this asset has now existed, the code language has kept changing, from LP1 to Java, Spark, etc. This is another crucial transformation aspect, necessitated not by client request but because of *technology change*. No literature was found reflecting technology change on reusable software assets.

A third transformational impact driving change is the pace at which *industry knowledge* grows, a pace the software asset must keep up with. Business knowledge forms the heart and soul of the asset. The asset owning team keeps their industry knowledge internally up to date using several methods, including learning or externally hiring staff with the correct industry knowledge. The reuser C1-I4 acknowledged this fact, saying the reuse gave her access to the asset plus a team of experts and the asset owners, confirming Ram & Devi (2019) as mentioned in Section 2.4.3.

The asset owning team explained that knowledge “is always a building” (C1-I1, L331). “Knowledge is never discarded. It may be reused at any time for a different type of an engagement” (C1-I1, L338). The team gave examples of when they made recent use of their almost 30-year-old industry knowledge. This hoarding and retaining of knowledge contrasts with some literature discussing the importance of organisational forgetting (de Holan & Phillips, 2004; Spender, 2008).

Innovation

The asset team described *innovation* as a “natural progression” (C1-I3, L125), constantly focusing on removing client pain by trying to understand the business problem and then asking, “Can we *repurpose the asset to address this issue?*” (C1-I3, L112). In this context, innovation happens gradually in small steps by building upon existing elements. This aligns with Fivet & Brütting (2020, p. 2): “Nothing is created, nothing is lost, everything is transformed”. It is an alternative to the radical revolution (Corrado et al., 2021). The interviewee used the word “repurpose”. Repurposing is defined as “minimal processing including some adaptation/additions” and goes beyond reuse which foresees “negligible processing” (Rose, 2019, p. 62).

The IT provider is one of the *patent* leaders worldwide. However, the asset team does not tie asset innovation to patent release. “Yes, I have *patents*, but not in anything to do with the work I do on our assets” (C1-I2, L395). The interviewee went on, “we didn’t benefit anything except ... we’re looking to protect ourselves with that. ... And I think that’s ... interesting, but we don’t do that. We kind of don’t have time” (C1-I2, L408). A reason for not patenting the reusable software asset could be due to spillover in the market (Haskel & Westlake, 2018). A patent would not stop other IT providers from competing.

Automation

Automation is a trend in software asset reuse. As industry knowledge became more sophisticated, the asset team adapted their offering. They realised that their clients had a strong “investigative skill set” (C1-I1, L353) but “didn’t understand data” (C1-I1, L361). “We were automating the tool ... so that they were just getting the answers they needed. So, now we see it more of an outsourced type. ... we run the ... monitoring on the transactions and just give them a report back and tell them what they need to” (C1-I1, L364-366). As technology advanced, the asset “went from being an on-prem offering to now a solution as a service” (C1-I1, L348-349). To explain this further, the asset developer elaborated, “I envisioned the future of where we become a service that a customer can procure independent of our consultants.” (C1-I2, L455). Section 2.3.1 discussed the shift from <creating functionality and charging for it once> towards <owning and providing functionality and charging clients for the services on an ongoing basis>. Automation on the provider side helps reusable IT services to be delivered efficiently.

Cloud

Cloud eases the deployment of reusable software assets. “There’s a lot of benefits to a cloud deployment SaaS-based offering. One is ease of deployment ... We’re able to take the client’s data run ... as a service offering. ... It increases our ability to close the contract with the client faster because we can prove the ROI faster ... it decreases the reliance on the customer’s internal team or audit team, ... because we increase their efficiency by doing it for them.” (C1-I3, L207-220). The asset team is just at the start of exploring the benefits of the cloud. One advantage is clear already: offering reusable software on the cloud saves deployment effort.

The shift to the cloud can be seen as a step towards the circular economy model when it comes to reuse, but it breaks with access conventions. To reuse something on the cloud, the asset owners must give the DevOps team *access to the asset*, “the role of read on certain repositories that they need if we decide to kick off a build in the DevOps tool” (C1-I2, L95-97). Before that, the access to the asset, “categorised as [a] crown jewel” (C1-I2, L19), was solely with the asset owners. Atasu (2021) uses the term access, too, to refer to what Birch & Muniesa (2020) name control. In a reuse situation, it would be the role of the DevOps team to adapt the reusable software asset on the cloud to the new reuse situation. The effect of shared development on the reusable software asset still needs to be determined.

Security impacts the reuse setup. “Our clients are still very nervous about using a cloud environment ... the data privacy issues, and the amount of data that we need, or the detail of the data that we need in order to run the analytics. So that's one area that it's just their concern. ... That's a negativity on the cloud” (C1-I3, L226-228). There are security aspects to consider, “... having dashboard views back to them and then we can consolidate more data that way. We can bring in external data. You can't do that ... when you start doing [...] analytics because of the level of security that you need. So, each of those databases have to be protected. So, the one drawback ... is that we have to *stand up a separate cloud environment for each and every ... implementation that we do*. ... We can't just put them on a different floor because they can't be sharing databases. They can't be sharing the same engine due to the privacy” (C1-I3, L237). For Case 1, it can be concluded that showing different asset versions (one per client) in Figure 2.16 (II) applies both for on-prem and cloud installations.

4.2.3 Case 1 Asset Distribution

Figure 4.7 presents the distribution topics raised by the interviewees in Case 1. These will be discussed now.

Discussion topics	Case 1
Asset distribution	
Internal marketing	...
Offer	...
External marketing	...

Figure 4.7 Discussion topics of Section 4.2.3 (source: author)

Internal Marketing

In traditional IT consulting, consultants become aware of a pain point, discuss it with their client, capture it and a team develops a solution. For reusing an asset, knowing the client references of the asset is not sufficient. “What's most *important is the use cases and not necessarily the client reference* per se, but the use case or reuse of the asset. So, being able to kind of share what we have done [giving examples]. And what some of those use cases are, and the recovery amounts that we have seen, or realised for the clients that we've performed these engagements. For it's really essential, ... for me to be able to walk in and say, let me give you an example ...” (C1-I3, L190). Only then can consultants phase in existing assets for new reuse. The use case needs to be communicated and made centrally available, e.g., in the asset repository holding meta-information on the asset. Typically, these repositories state what the asset does. They do not state the use case, i.e., what client pain points the asset addresses. This is an aspect not covered to date in the literature.

Clients stay for ten years or more just because of the asset. The asset team provides ongoing services and maintenance for long-term clients. The existing client base must be expanded via internal and external marketing to grow.

All assets start small with a clear scope in one domain, focusing on the vertical reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010) as described in Section 2.5.5. Because, compared to a product, there is typically a smaller set of clients, the asset owners look for ways to expand the client base, applying horizontal reuse and benefiting from assets being a bet into the future, as described in Section 2.2.4. Over time, effective marketing lets the asset team find new clients in areas not initially targeted.

While asset marketing remains a challenge, “it's about 50:50” that a reuser finds the asset team versus the asset team finds a reuser (C1-I3, L80). This means in half of all cases, the reuser is reaching out to the asset owner when there is a need for the asset. This confirms that pushing the asset into the market is important, as described in Section 2.4.7 (Wallace, 2018). Tremendous communication spreading the news on the reusable software asset is required to generate a company-internal pull effect.

Since IT enterprises are constantly transforming, the asset team has become distributed across the organisation over the years, reporting to different managers and working in

other locations. The business success of their asset virtually ties them together, but they have to justify their existence continually. Typically, they are part of the organisation where the asset was initially developed. This setup may provide little support and sponsorship for the asset. Therefore, some of the asset team members moved into the asset organisation. “With the *asset organisation* now, with some of our developers, even reporting over there, ... that has added so much value to our asset. ... we've all been working together over twenty years, but I think more from ... the aspect of all the marketing-support we're getting now, and the ... active involvement and ... the asset community itself has just grown considerably. And just even the monthly asset calls ... provide so much value” (C1-I1, L228). The *asset organisation* is the only central place for sponsorship and guidance for the geographically dispersed asset team in Case 1.

The asset owner C1-I3 gave examples of further marketing activities. She runs an *asset-specific slack channel* which externalises and channels written, internal asset reuse-related discussion and makes it searchable, confirming the literature findings in Section 2.4.3 when from an knowledge work context individual tacit knowledge is consciously transformed into explicit knowledge (Dreyer & Wynn, 2016). The asset is featured in the *company's intranet*.

Offering

Business opportunities can be mapped against pre-defined standard offerings. This creates a powerful logical link enabling an asset to be pulled into an opportunity at the right time, the window of reuse (Wolfram et al., 2020). In the pilot case, the reusing team in China mapped their business opportunity against a standard offering and then looked up the assets supporting the respective offering (C1-I4). This is how the reusing team became aware of the reusable software asset. *Mapping assets against standard offerings* allows reusers to find a suitable reusable software asset more easily.

Sometimes no standard offering is available for the asset to be mapped against. The mapping is non-trivial due to dynamic changes: in line with the dynamic capability theory (Marrucci et al., 2022), assets and standard offerings come and go. Communicating these changes across a large global IT enterprise remains a challenge. The changes are not only industry- but also technology-related. So, the asset “went from being an on-prem offering to now as a solution as a service” (C1-I1). No literature could be found on mapping reusable software assets against standard offerings.

The asset reuser looked up the asset in a *software asset catalogue*, as AL-Badareen (2021) described in Section 2.5.3. This asset catalogue is aimed at the non-technical staff and maps each asset against pre-defined offerings.

The asset owner C1-I1 conducts company-internal, industry-, service line-, and geography-specific *sessions on her asset* and its capabilities. Additionally, she scans business opportunities logged in the company's opportunity system and uses her personal network to identify clients needing her asset. There is again a time aspect to niche marketing, a 'window of reuse', during the reuse process (Wolfram et al., 2020). This reduces the prospective set of experts willing to listen to the asset presentations. There are internal experts for whom the asset information comes too early as they are not yet aware of their client needing the asset and internal experts for whom the asset information comes too late because they have chosen internal development instead of reuse; these client accounts have no interest in attending the asset presentation. Finding a client account needing the asset at the moment while constantly growing the asset's capabilities is like hitting a moving target.

External Marketing

The software asset needs to be actively presented to potential reusers. "*Marketing, marketing, marketing ... I am in a niche area. I'm only allowed to focus on certain opportunities. I take it upon myself to break out of the industries to go cross into telco and others ... our asset is ... so cross-industry. It's not niche, and so we need to be able to sell across [name of the global IT company] as a whole, and not just in the [sector a] or not just in [sector b] ... it has to be an asset that can be sold across [name of the global IT company] and I think where we get ourselves into trouble is that we have these asset owners sitting in selective narrow organisations and not sitting where we have the ability to go cross [name of the global IT company] as a whole*" (C1-I1, L428-437). While the interviewee could present to external clients, she often does not have the contacts and permission. This is the case for clients outside the asset's current domain.

The interviewee suggests changing the organisational setup of the asset owning team. If she and her team were not part of a specific, narrow domain organisation but centrally hosted, contacting external clients of diverse industries or service lines could be easier.

Until then, to implement a horizontal, cross-domain reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010), the asset owner has to apply a *two-step sales process* by first proposing the asset to the company-internal client account team and, after receiving contact details and permission, presenting the asset directly to the external client. The implications of this have not yet been mentioned in the literature.

The asset owner is *doing marketing, but she is not part of any marketing team*. “What I've been doing, is reaching out, working with the industry leaders to get them adopt the solution ... But it's a hard sell. It's difficult to do that. You have to be your own person to go out and market and sell these assets because we're not sitting in the asset organisation per se, and we're trying and we're not seen in an industry per se, cause we're really cross industry for our solution. And so, it does give you a bit of a challenge sometimes” (C1-I3, L272). The interviewee is not a typical member of her department. This affects the support she receives for her mission.

External marketing activities include “reaching out to ... getting actively involved in the communities, whether it's ... our existing client community and doing webinars ... to tell them what the latest releases or the ... new use cases we have. Or it's participating in [name of an external industry trade show], and our [name of an external industry domain] Association ...” (C1-I3, L280). These external marketing activities give the asset reuse team direct contact with external clients without involving company-internal account teams.

Information is externally available on the asset, including subtitled videos on *YouTube*. The asset owner mentions the asset on *LinkedIn*, especially after attending *external conferences* or giving webinars for the client industry audience. Two of the IT company's *client service centres* share information and demos on the asset in two different locations. The asset team has received *internal and external awards* for their asset. Occasionally they issue *newsletters to their external clients*. They have also *streamlined their pricing, adopting a simplified pricing model*. *Support services* are offered for the asset, typically 8 am – 5 pm local time, depending on the country. This is a heavy but critical workload accomplished by one person. When C1-I3 is on holiday, there is no marketing for this asset.

The interviewee concludes, “we definitely need to do more marketing around our offering and commercials ... That, I think, would provide a lot of value. I don't think we ever had

the budget for marketing ...” (C1-I3, L427-428). She decides to work harder. In the past, they *tried assigning a dedicated marketing person or seller* to selected assets. Unfortunately, this has proven difficult in practice due to the small size of the asset not justifying a dedicated person, the wide and deep reach to the clients this person would need, and the agile nature of the asset making it challenging to keep up with all the many asset updates implemented for an increasing set of diverse clients.

4.2.4 Case 1 Asset Reuse

Figure 4.8 presents the asset reuse topics the interviewees raised in Case 1. These will be discussed now.

Discussion topics	Case 1
Asset reuse	
Asset benefits	...
Reuse	...
Abstraction	...
Systematic reuse	...

Figure 4.8 Discussion topics of Section 4.2.4 (source: author)

Asset Benefits

One interviewee explained, “the special value that our asset gives ... is that we are going in there with an offering again that can be reused. So, it lowers the development cost ... it accelerates our ability to quickly do a [Proof of Concept], and do an implementation for a client, because we have that plug and play capabilities, but also that flexibility of the configuration. It also opens up the opportunity with that client base to start in one specific area and then ... ask them [to reuse] the asset across the enterprise for different use cases” (C1-I1, L460-465). This statement lists *benefits* but is too euphoric. Pure reuse, as such, has lower development costs. On the other hand, a significant administrative effort must be funded until reuse can occur. Further, every reuse requires a certain amount of adaptation. A demo may be plug-and-play – the reuse of a software asset is not. Configuration flexibility means that one opportunity with the client begins in one specific area. Later, the same asset is leveraged across the enterprise for a slightly different use case.

Software asset reuse “*makes our risk rating so low in contracts, because ... we’ve proven it over and over and over again*” (C1-I1, L506). Low risk-rating is another key benefit of asset reuse which could not be found in the literature.

Further, it would be impossible to generate the same amount of *revenue in a niche market*. Custom-developed software would be too expensive, and a software product would not resonate with the clients. This has not yet been covered in the literature.

Reuse

The concept of *reuse* can be applied to the entire asset as well as to components of the asset which “can be reused, but not as a formalised component or a widget” (C1-I2, L267-268). However, if “you want to develop a repository of ... reusable components ... it has to be funded” (C1-I2, L281-283): any reuse requires upfront investment. Exploiting reuse to the max by using reusable components within a reusable software asset leads to increased complexity due to additional version dependencies. This confirms KPMG’s (2018, p. 15) “Try to reuse ... as a whole”-principle as explained in Section 2.5.5.

Abstraction

The asset owners described their asset as flexible and suitable for multiple industries. Reuser C1-I4 explained how she abstracted the use of the asset to try and solve her specific problem in China. Section 2.5.3 describes how Sandhu & Batth (2021a) discuss abstraction in connection with asset storage, how abstraction needs to be factored into reusable components to increase the chances of reuse (Biggerstaff, 1994), or is generally required at the time information becomes knowledge (Burciu & Kicsi, 2015). This case demonstrates that *abstraction is needed to brainstorm a new use case*, a topic yet to be covered in the software reuse literature. This addresses the point raised in Section 2.3 of how experimental situations are related to one another – they are connected through abstraction. This case also shows that the decision to reuse is typically made in the early phases of a process (Jonsson et al., 2021).

Systematic Reuse

All three interviewees were asked how systematic reuse can be explained. One interviewee summarised, “we have a very standard approach. That’s *very systematic that we’ve used for over 100 clients*. We just follow the same process” (C1-I1, L505). The

interviewees did not provide an answer detailed enough to replicate it to new situations. It takes more to systematically reuse.

Figure 4.9 shows the timeline of the reuse processes for Case 1 described in Section 4.2.

Case 1 Process Flow

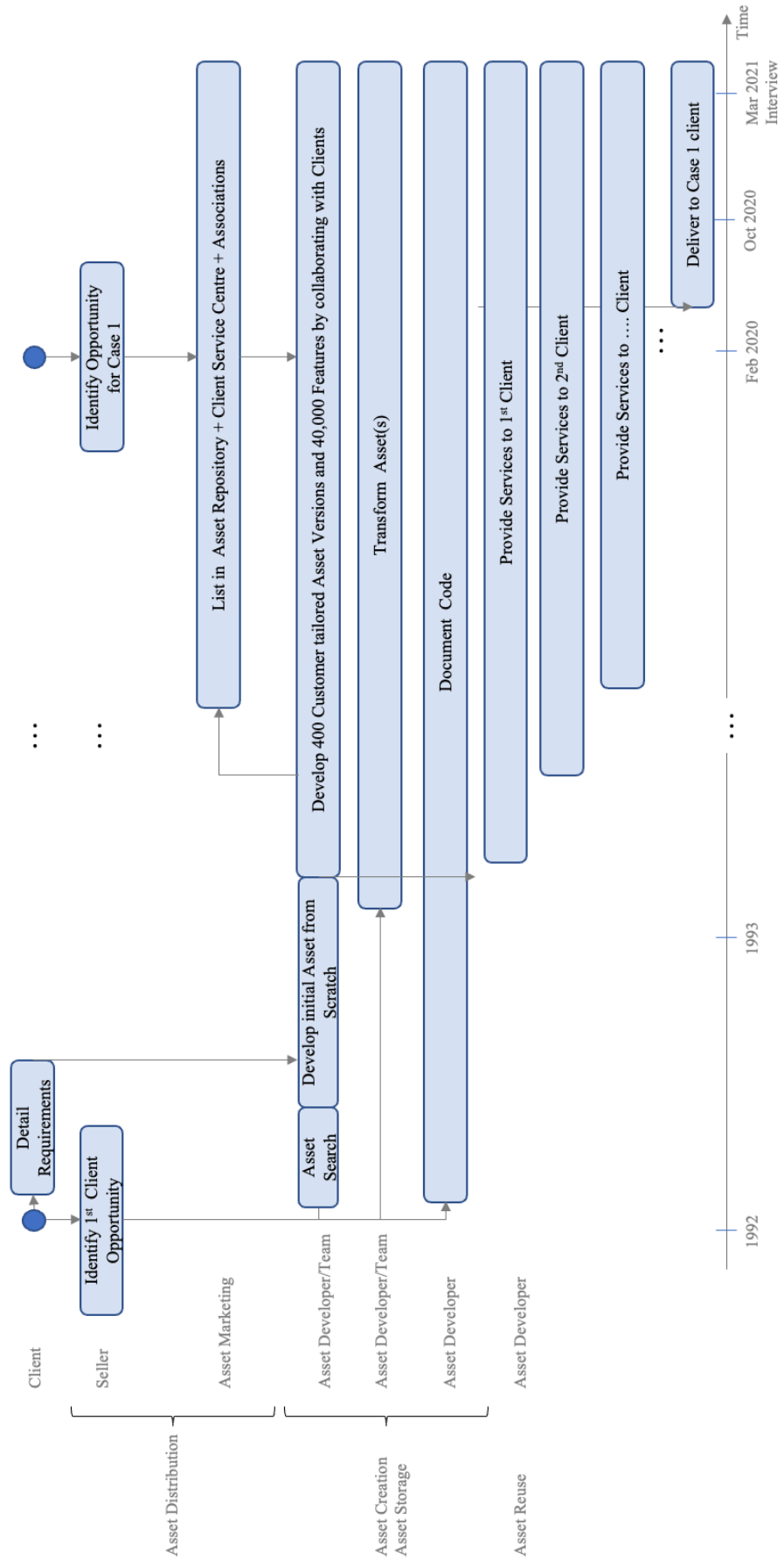


Figure 4.9 Reuse and process timeline for reuse Case 1 (source: author)

4.3 Findings Case 2 – Reuse Initiated during Sales Generating Revenue

Case 2 covers the reuse of a multi-tenant software platform asset. The idea for the asset originates from senior experts in global and North American roles. The asset was, and still is, developed by geographically dispersed teams in the US and Europe. The lead seller of the asset is in the UK. As indicated in Figure 4.10, the featured reuse Case 2 takes place at a Belgian bank and has country-wide effects within this bank. All bank employees make use of this HR-related software asset. This sales-driven asset reuse generates revenue through a licence and service fee.



Figure 4.10 Case 2 - Location of asset owner versus asset reuser (adapted from (Geology.com, 2021)

4.3.1 Case 2 Asset Creation

Figure 4.11 presents the asset creation topics that the interviewees raised in Case 2. These will be discussed now.

Discussion topics	Case 2
Asset creation	
The asset itself	...
Asset team	...
Funding	...

Figure 4.11 Discussion topics of Section 4.3.1 (source: author)

The Asset Itself

Creating the new software asset was a game changer. At this time, there were “hundreds” of software systems already existing in the market in addition to quite a few company-internally. Some of these internal software systems were fully customised to local conditions. The original thought leaders behind this asset shaped a team of decision-takers with “executive vision and commitment” (C2-I1, L197). The two of them, supported by further experts, e.g., developers, looked at their task from a holistic, worldwide corporation perspective (C2-I1). They not only created a software platform that fitted future needs but took it to a higher level by also handling the processes before and after the software’s core mission (C2-I1, C2-I3). Although there were previous and still existing solutions, the team “started to develop from scratch” (C2-I1). Developing the asset from scratch allowed the asset team to directly address the biggest pain points. They made their asset the *one place to go* for the user and management. It included artificial intelligence (AI) and complex analysis (C2-I1). They expanded the scope and set things in perspective by bringing in additional features and functions from microservices of a related software asset.

Originally, the asset was developed for company-internal use only (C2-I1). Later, a team of involved executives decided to have a joint investment to turn the newly created internal software into a software asset that could also be *offered to external clients*.

The team of decision-takers was familiar with the existing solutions in the market. They scanned the field from a content and technology perspective (C2-I1). Each decision-taker was in that business for more than ten years. As for Case 1, when they brought this new asset into the commercial market, they were sure it did not yet exist (see (1) in Figure 2.17) (Mateen et al., 2017).

The software Asset 2 went live in 2016. Its commercialisation process started in 2017 and took one year. Since 2018, the asset has followed a SaaS model (C2-I1).

One instance of the software simultaneously supports multiple customers (C2-I1). This characteristic is typical for cloud software but unusual for a software asset, according to Figure 2.16. It would rather indicate a software product. On the other hand, interviewee C2-3 explained that the asset is seen as not mature enough because it is not yet an easy plug & play product that can be sold “through the channel”. The interviewee admitted

that the asset is “too complicated for that” (C2-I3, L62). Special services, consultancy and customisation (C2-I3) tailor this cloud asset to fit niche areas where no product exists.

Software assets benefit from *synergies with other software assets*. In Case 2, the asset is linked “into a broader platform together with other assets” (C2-I3). This confirms Haskel’s & Westlake’s (2018) synergy-finding mentioned in Section 2.2.2. The interviewee sees further integration as a requirement because the platform solution, including all assets, is managed by a different team. This setup makes it hard to manage (C2-I3).

In contrast to Case 1, there is currently *no internal competition* for the asset (C2-I1). This may be due to its complexity and wide-ranging functionality requiring major investment. However, external third-party tools emerged in the market around the same time. Their software is based on a similar principle. There are no plans to partner with any external providers (C2-I1).

Asset Team

In 2015, a new team was formed out of circa two or three geographically dispersed existing teams that historically worked on local solutions. These “*groups ... competed about everything*” related to this software asset (C2-I2). It took time to unite them and make them more friends than competitors.

Interviewee C2-I2 described how his team was historically working on a predecessor of the software asset. While his team remained as such, it *became part of the large asset team* that emerged and is now staffing one of several squads.

The asset development backend team comprises 15 developers and four dedicated testers responsible for quality assurance. There are circa 30 front-end developers and five quality assurance staff/testers. Further, approximately six to seven data specialists concentrate on relational databases. Five people also work on cognitive functionality. In total, the *development team comprises about 60 people* (C2-I2). The whole team is split into squads of a maximum of five people. Reuser C2-I4 estimates that circa 20 developers work on the extensions and functionality, his commercial client requires. Summarising, the asset team comprises design, development, services, and data squads (C2-I1), located in the US and Eastern Europe, and experts for selling and marketing the asset, based in the UK and Benelux. Consequently, the asset has a strong, historically developed, geographically

dispersed team. This setup confirms, in line with Section 2.4.3, that team-internal transactive processes lead to a logically organised knowledge (Zhou, 2019). There is credibility and trust among the team members (Human, 2020). The team's transactive memory increases the team's affective outcomes (Zhou & Pazos, 2020).

The asset owning team is very *stable*, unlike the fast-changing asset (C2-I1). “We are very ... ridiculous stable. We have some members like there is a huge significant core of people that have been on the team since 2015. ... We've grown since then, but even the people who joined in, in later years, we have very low turnover. ... The team is engaged, we're cutting-edge people like working on the team. They like the people that we're working with and so people don't leave, and it's not that people never leave. But ... compared to a lot of other teams ... we are much more stable. And part of it, I think, is because of the culture of the team. And ... we're winning and ... we have momentum, but we also have interesting technology. We have interesting problems, and we give people space to ... do the kind of work that they find fun and interesting as a technologist” (C2-I1, L367). The team stability of Asset 2 is in line with Section 4.2.1 about Asset 1.

Company internally, the asset represented in Case 2 has two strong *sponsors*, one from the business and one from the sales side (C2-I3). So, a widespread team of asset practitioners and their management is working hands-on on the asset. Additionally, a big team of executives is acting as stakeholders, talking on an executive level to clients.

The asset team embraces the agile principle, valuing people and interaction over processes and tools (C2-I1). This aligns with agile software development described in the literature and mentioned in Section 2.5.2.

Interviewee C2-I3 describes her team in terms of knowledge management (Wijnhoven, 2008): “We are like a *start-up* where we have a couple of clients” (C2-I3, L27). The team is starting to professionalise themselves, e.g., starting up a client service centre, planning to centralise their knowledge and communicating with the clients. They operationalise their work to lower costs while being more effective in the market (C2-I3). Reuse is typically in the hands of mature IT providers who produced software some time ago, which is now being reused. On the other hand, the term *start-up* is a good description of asset teams' situation. They are acting in an agile, entrepreneurial mode, as described in Section 2.5.2, experimenting with how to market their software asset. This aspect could not be found in the software reuse literature.

New business knowledge is incorporated into the software asset when the client issues new requirements (C2-I2). Further, asset-specific technical knowledge is built up gradually. Any new team members have to attend a boot camp (C2-I2) to get up to speed quickly. The boot camp covers both technical principles like shared libraries as well as business terms and logic. The reusable software asset acts like a sponge, absorbing all related knowledge from the owner and client side over the entire period of its existence.

In line with Case 1, learning plays an important role for the asset owning team in Case 2. “We learned a lot and we did a lot, and we didn’t stop.” (C2-I1, L333). The team picked up cloud technologies at the earliest possible point in time. They started with microservices very early, and as soon as Kubernetes came up, they adopted it. The asset team is very much into learning and applying new knowledge.

Funding

In contrast to Case 1, starting off this software asset as a big platform was a revolution. However, behind the scenes, this was based on multiple earlier evolutionary attempts, starting small over time. This confirms Hargadon’s (2002) thinking, as discussed earlier in Section 2.3. However, this big bang approach required substantial funding.

When the team started developing this asset, there was initially not enough funding to develop a cloud version of the asset. Creating the asset first for internal use and then finding paying commercial customers helped, from a commercial angle, to bring the asset into the cloud and fund the asset’s big development team (C2-I1).

Adding to what interviewee C2-I3 said about working like a start-up and what interviewee C2-I1 called fun work and interesting technology, interviewee C2-I2 described how developers are motivated by business needs to introduce new functionality and innovation. The team is eager to respond to any requirement the client has, enabling the client by removing the client’s pain points. In addition to being helpful to the client, the second motivational aspect is the team’s drive to use new technology. “This is like freedom ... which we get” (C2-I2, L264). Interviewee C2-I2 explained they have the *freedom to try new things* and are not blocked.

Total *freedom* in choice of technology could lead to using too many different technologies. Interviewee C2-I2 described, therefore how they introduced the four-people-rule which means there should be at least “three other people who have an

advanced knowledge of the technology” (C2-I2, L383) in terms of configuration, usage, and sharing with new developers or stand-ins. This demonstrates that the freedom within the asset team is bigger than for regular teams. Further, any new choice of technology needs to stand the security demands of the clients (C2-I2), i.e., it must be documented and comply with the law relevant to the clients. The motivation the asset owning team gets out of the freedom to try new things is another indicator of autonomous job motivation as explained in Section 2.4.7 (Bidee et al., 2013; Frakes & Fox, 1995; Gagne et al., 2019; Karim & Majid, 2018). Agreed rules ensure that the freedom of choice has positive effects.

4.3.2 Case 2 Asset Coding and Storage

Figure 4.12 presents the coding and storage topics raised by the interviewees in Case 2. These will be discussed now.

Discussion topics	Case 2
Asset coding and storage	
Agile development work	...
Innovation	...
Cloud	...
Code conversion	...
Patent	...
Knowledge	...
Forgetting	...
Security and certifications	...
Partnering	...
Automation	...

Figure 4.12 Discussion topics of Section 4.3.2 (source: author)

Agile Development Work

The asset owning team applies the agile work model. Interviewee C2-I1 described how user feedback impacts asset development. “We do a lot of user testing. *We listen very heavily to user feedback* and ... prioritise feature function ourselves” (C2-I1, L254). The asset team works with the various user groups to ensure their requirements are mature enough for the next planning and implementation step addressing the topic of knowledge management (Wijnhoven, 2008). User feedback is monitored by the operations teams for *continuous improvement* actions (C2-I2).

To benefit from these client-induced software improvement ideas, the software asset developers follow the *agile* methodology, “being flexible on what’s next” (C2-I1, L248). The asset owning team has two levels of requirement planning (C2-I1). They have significant goals they want to achieve in the first half of the year and future goals for the second half of the year. From a sprint planning perspective, they operate three-week sprints, i.e., every three weeks; the clients receive new feature updates and functions, applying automation for continuous implementation and deployment. The asset development team can deploy new code circa five times a day (C2-I2).

There is wide flexibility in developing the future asset (C2-I3). “We can go into all kinds of directions, because the wish list of features and functions is very long ... I think it’s important that we make some *strategic decisions* on where to put our investment. ... We can’t go in each and every direction and build it from there. ... So, it’s different swim lanes. We need to increase our client base, and with that, we can re-invest into enhancing our products. We can enhance our services.” (C2-I3, L233-235). Interviewee C2-I3 twice used the term “We’re really a start-up”, adding that much work is ahead and the market is moving quickly. She means that broad flexibility is a chance and a curse. The future business success of software assets depends strongly on pertinent strategic decisions. However, there is rarely a consortium making strategic decisions. Instead, the future of the asset could be left with the individual, as detailed in Section 2.1 (Younoussi et al., 2019). This could undermine systematic reuse and lead to ad hoc successes. Garcia (2007) demands a clear reusability strategy which is required to secure market advantage (Abualoush et al., 2018; Khavandkar et al., 2016) in line with proper knowledge management (Wijnhoven, 2008).

Innovation

Per Section 2.5.6, innovation is a key driver for reusing a software asset. Interviewee C2-I4 (L219) said, “innovation comes ... from both ways, in a sense that typically the asset owners need to bring the new features that might be beneficial for the customers. But the partnership that we have in place and all the workshops, they are using a bit design thinking workshop with end users going through the screens and getting their feedback while providing then a user experience how it would look like once the screens are being developed. These provide a lot of information, which is helping us to further improve the assets and bring innovation as well.” Both the asset owners and the partners strive for innovative software asset improvements. Here feedback is of the essence (C2-I2).

Sometimes this feedback is gathered during community calls (C2-I3). Summarising, the *innovation for Asset 2 often resulted from received feedback*. None of the interviewees found reuse counterproductive to innovation as Section 2.5.6 mentioned and Stenholm (2019) has claimed.

Interviewee C2-I4 stated that an important *decision criterion* for his client signing the multi-year contract on Case 2 asset reuse was the price and the partnership model on the software asset.

Reuse has integrative power. “Innovation comes from the dynamics in the markets and the rapid changes but also how we distinguish [ourselves] from the competition. We claim to be the integrator, ... to have an open architecture, a modular architecture” (C2-I3, L103-104). The interviewee explained how they provide platform services that can be customised by switching microservices on or off – an example of how an innovation is transformed into a practical application (Baptista et al., 2015), as discussed in Section 2.5.6. The development is very dynamic, confirming the dynamic capability theory (Marrucci et al., 2022). Reuse is not using an existing item one more time (C2-I4). Instead, reuse integrates new ideas from the client, the competitors or the asset owning team thus expanding a current software solution innovatively.

Cloud

The asset featured in Case 2 is cloud software. This has pros and cons, given the agile nature of the software asset. Interviewees described implicitly why the software is an asset and not a product. In this context, interviewee C2-I4 gave examples of how they sunset one software feature function, which impacted all clients. In contrast to Case 1, where each client has an individual cloud setup, there is one for all clients in Case 2. Due to this *one cloud setup for all clients approach*, any function change immediately affects all the clients, which is not always easy to communicate (C2-I4). “It brings additional complexity for the development of new extensions ... when we change our strategy ... and often customers need to pay for that as well” (C2-I4). Even if there are no charges, “it brings additional effort for them to make sure that they can migrate the whole environment to a new solution” (C2-I4). The one cloud setup for all clients-effect is much stronger for software assets than for software products since software assets are much more flexible and agile; they change very often, at least once per reuse (Lear, 2021), as discussed in Section 2.5.5.

In Case 2, interviewee C2-I2 stated that every client gets *the exact version of the asset*, “when we, for example, decide to move some button from the top left corner to a top right corner, it appears for every client immediately” (C2-I2, L210). Further, the team maintains no related assets in the same space (C2-I2). However, interviewee C2-I1 (L377) added, “sometimes we only turn on functionality for some clients and not for others. We try to minimise that. So, in general, if we turn off functionality, we turn it off everywhere, or if we turn it on, we turn it on everywhere”. C2-I1’s comment is in line with C2-I3. The client is provided with a *configured solution* – “is not just bringing a product off the shelf to the client but really looking at their needs and to see how to which extent we can help them” (C2-I3). The one-cloud setup for all clients-approach is a trade-off. It simplifies operation and maintenance. At the same time, it complicates future development and asset marketing. In practice, a one-cloud setup can only run with configuration.

Running just one code basis across all clients is unusual for an asset and challenging. The asset owning team, therefore, extended the asset by plug-ins and by connectors. *To achieve systematic reuse, the development team has put a “Tenant Operation Team” in place* made up of specialised developers. They support the clients by explaining to them the features of the API backend services and tools. Further, they develop integration tools for the client. They act as a customisation team by customising the plug-ins. In doing so, the “Tenant Operation Team” (C2-I2, L399) ensures the asset’s best possible reuse. They work with each client’s IT team and explain certain functions to them, preventing the client’s team from having to study the entire API documentation. While about 60 developers work on the core asset, ten of these customise extensions and connectors and directly communicate with the client on the code level. These ten developers also develop new extensions if needed (C2-I2). Summarising, the one code base can only be rolled out to a small, diverse client set with the help of plug-ins and connectors. Alternatively, each client would have required their features and functions as in Case 1.

Code Conversion

Code conversion keeps the asset owners busy. Although the asset in Case 2 is much younger than that in Case 1, interviewee C2-I2 reported similarly how code was in Rational Team Concert and then migrated to GitHub. Some of the code is inactive today but can still be checked in the old system. Within the six years the asset has existed, it has been moved three times *from one environment to another*. Right now, it is on the public cloud. Moving the code took a lot of time and technology changes are required additionally. “Every year there comes some strategic decision to change something or introduce some new standard, which requires all the teams front end, and back end, and testers to sit together and really rethink the ... technical direction of the whole product. And this is also useful ... we regularly, thanks to these pushes, move to a new environment or use new technology” (C2-I2, L610). The asset team is constantly evolving, and this motivates the team. Reuse strategy and knowledge management overall are important and highlighted both by interviewee C2-I3 earlier and now by C2-I2.

The *speed of technical change* has increased. “The way how they developed a thing five years ago, total differs from how they would write it nowadays. So, even ... those things rust, ... need some refreshments or refactoring.” (C2-I2, L535). In line with what interviewee C2-I2 said, interviewee C2-I1 described how the asset moved from running microservices on DB2 using Cloudant to using the then-just-emerging Kubernetes on the cloud. Accordingly, there is rapid technical change, and it affects all areas, including reuse and documentation.

Patent

Patents have no ubiquitous power. In contrast to Case 1, Case 2 has a few patents in place for this software asset (C2-I1). The asset team tried to file patents on this asset in China but failed. They fear a copy of the asset could emerge from China (C2-I1). As shown in Figure 2.6 in Section 2.2.2, asset strength does not always correlate with innovation. The danger of spillover in the form of unwanted copies counteracts the superior chance for synergies.

Knowledge

The software asset contains knowledge on SaaS, microservices, Kubernetes, business knowledge, and test and deployment automation. Further, the asset is a mobile-first

application, i.e., although it is not a mobile application, it acts like a native mobile application so that users can add it to their phones. The asset has been set up to scale dynamically up or down to handle various capacity challenges (C2-I1), supporting the dynamic capability theory (Marrucci et al., 2022). Interviewee C2-I1 is proud of the innovations achieved on the technical side but also on the business side by providing a platform which acts now for various persona groups as the one place to go. Expert knowledge enabled innovation which finally opened up new markets, as explained in Section 2.5.6 (Tidd & Bessant, 2020).

The asset has “grown in features” over time (C2-I1). The team has “been developing continuously since” it started in 2015. This demonstrates Lear’s (2021) video asset-related comment in Section 2.5.5 that assets are constantly changing and is also relevant for reusable software assets.

Forgetting

Actions are taken so *no code will be forgotten* (C2-I1, C2-I2). The developer team had an API as a communication protocol to access the functionality and code of the previous system. However, over the past five years not a single client has made use of it. They, therefore, now prefer an archive which requires no maintenance. “Sometimes in the past, there happened that some functionality ..., actually was very similar to something else what was desired. So, we could basically copy paste and adjusted the code for totally different purpose because it was such similar. ... We don’t throw away like that ... We’d rather put it to some archive to some storage place. And if we find a user for something different, we may reuse it. We may recycle, but sometimes it’s lying in that storage almost forever” (C2-I2, L560). In line with Case 1, the asset owning team in Case 2 actively prevents forgetting any knowledge. This is in contrast with Spender (2008) as in Section 2.4.5.

Security and Certifications

There was a significant effort in implementing the *security* and privacy required by the switch from internal to external reuse (C2-I1). The asset owning team staffed an information security squad and ran a formal security audit to document and resolve various security constraints. It turned out the asset was well protected from external threats but not from internal threats such as sabotage or theft by staff. Addressing the

internal threat took about a year. As for any software, security is important for reusable software assets.

The software asset became *ISO 27001* certified, an industry security certification. To achieve this, the asset team received support from a company specialist in ISO 27001 certifications. This certification helped the clients' security reviews of the software (C2-I1).

Interviewee C2-I1 (L230) summarised the importance of the security audit and ISO certification: "We didn't realise how important that would be for commercial sales, especially to sensitive customers". This is a fact that interviewee C2-I4 closely confirmed when discussing the topic cloud. Security and certification are both critical for external reuse and should be started earlier in the process (C2-I1). Neither topic is covered in the literature.

Partnering

Initially, the account team proposed an implementation approach to the client where they wanted to develop the required features, but the price was too high for this client. That's why the IT company's account team developed a partnership model "to find a way to reduce the implementation costs ... where the new features are used to further improve and further develop the assets that we are using. ... The customer would then don't have the intellectual property of the solution itself" (C2-I4). Consequently, *some of the content provided on the platform asset has been acquired from a third party* (C2-I2). Further, the asset team is partnering with other company internal departments (C2-I2). This benefits both the client and the IT provider from a price and functionality perspective. By sharing one asset, all clients *get greater functionality* than they would for the same amount of money invested into an individual development from scratch (C2-I4).

Automation

The developer team follows a "fast deploy, fast recover" strategy (C2-I2, L211). This means new code will get rolled out quickly. The time lapse between deployment to the test environment and deployment to the production environment is about two hours. *Automation* is applied during deployment and testing (C2-I2). If something does not work as expected, the code issue will be rolled back and fixed quickly.

4.3.3 Case 2 Asset Distribution

Figure 4.13 presents the asset distribution topics raised by the interviewees in Case 2. These will be discussed now.

Discussion topics	Case 2
Asset distribution	
Client support	...
Two-step-sales system	...
Request for proposal	...
Operative marketing	...
References	...

Figure 4.13 Discussion topics of Section 4.3.3 (source: author)

Client Support

Client support matters. The industry expert C2-I3 is driving the setup of a *client service centre* in one capital city in Eastern Europe. “You need to take responsibility for the *client support* and the offering – the asset owners don’t do that” (C2-I3). Since this platform asset is co-supported by various individual assets, the asset owning team is not taking the lead as it typically does. Instead, an additional management layer is required as a support function. Until that is put in place, the industry expert is “doing it because ... it’s ... a must do at this stage” (C2-I3). Many clients expect the services provided for software assets to be like those of software products.

Service centres can be used as a communication platform. “We are starting up a *client service centre* now in [name of a city] because we need to centralise our knowledge and our expertise to standardise as well as communication to the clients” (C2-I3, L28). Standardising processes increases the efficiency of the software asset owning IT company.

Two-step-sales System

Interviewee C2-I3 described a *two-step-sales system*: “we actually marketed only through our own practitioners through our own sellers”. Because clients are tagged to organisational units, the asset’s marketing expert only has direct access to a maximum of one client set. Consequently, internal enablement sessions to interested practitioners, industry specific groups, and community sessions are being run. “We could do much

better on the external marketing and communication strategy to be part [of] more heavily interest groups, communities externally. But I definitely see that we have a bandwidth issue and, and a budget issue to, to support it” (C2-I3). One reason could be the underlying assumption that reuse would set in automatically so that no marketing is required.

Interviewee C2-I3 is running an internal website on the asset, involving a weekly enablement session. Every two months there is a slot to discuss the sales status of all asset opportunities as a team. “There is a proactive reach out to account teams and industry teams ... So, there are multiple layers into *the internal enablement*” (C2-I3). Marketing happens predominately internally. “We mentioned [the asset] in lots of articles from our own executives, calling out our great success. ... But we are not necessarily doing external marketing on [it]” (C2-I3).

The achieved asset revenue covers a small percentage of the efforts of the wider asset team and flows mainly into asset development. There is *no dedicated marketing budget for the asset*. To fund some marketing, other organisational budgets need to be tapped (C2-I3).

Request for Proposal

Interviewee C2-I3 writes geography-wide on RfPs. Sometimes, she identifies asset opportunities herself. Thus, interviewee C2-I3 is involved in a lot of *client leadership sessions*, especially in the early stages of an opportunity.

The asset kicks off transformation as a result of client conversations. “We take them on a journey ... and often an RfI [Request for Information] or RfP is coming out of [it].” (C2-I3, L145). This requires the reuse discussion to start *very early in the sales cycle*.

This particular asset has been marketed via a *platform solution* in conjunction with related assets, not individually (C2-I3). The integration of this asset and its peers into the platform solution is seen as essential. This confirms that intangibles benefit from synergies with other intangibles as discussed in Section 2.2.2 (Haskel & Westlake, 2018).

The asset person in charge of finding new clients works on “*lots of opportunities* in different sales stages” (C2-I3). Most of these will never materialise for various, often non-asset related reasons, e.g., decision-making processes at the client side. So, it is a challenge to focus on the opportunities that have the biggest potential.

Interviewee C2-I3 mentioned how the team is involved in *answering Requests for Information* (RfI) that focus on the asset's target area. The challenge is that the client will decide on the provider based on the lowest provided price. "It will be very difficult for us to win, because then it will be ... based on price" (C2-I3, L140). This indicates that reusable software assets are not as price-competitive as originally assumed. Reasons for this can be high administrative costs for creation, storage, marketing, and finally reuse of the software asset. Both C2-I3 and C2-I4 comment independently from each other on the high price/cost of the asset. This finding contrasts with previous literature findings (Barros-Justo et al., 2018; Krüger & Berger, 2020; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Sandhu & Batth, 2021b; Van Buren et al., 2016). A mature, historically developed and therefore high-quality software asset can, given all the required reuse-related activities, be more expensive than a simple, quickly put together one-off software.

Building and maintaining a software asset of this size and complexity is costly. "We prefer actually to come in from a transformation perspective where we are part of a bigger transformation" (C2-I3). Therefore, the chances of winning *big, single source engagements that focus less on a specific software function but on overall process transformation* are higher, if the client gets to the point of deciding on the asset. "If we are going head-to-head with our competition, we are too expensive, and we'll lose" (C2-I3). The savings are counter-weighted by the high cost of maintaining the asset until the next business opportunity is won. This may explain why asset teams are busy shaping new opportunities rather than trying to map their asset against opportunities emerging via client RfP.

Operative Marketing

In line with the lack of marketing budget, the operative marketing of the asset is left, in this case, to one individual. It is challenging because there is always only a small target audience. To address the issue, this asset team has developed a unique approach – they engage in *analyst calls*. Due to the excellent asset services and technology, analysts list the asset team as thought leaders in their analyst reports which go out (C2-I3). This helps in external asset marketing.

The asset team has introduced another marketing tool, *client community calls*. Here all of the – few but big – clients join one virtual session to learn from each other about making best use of the asset. "They were very eager to speak to each other" (C2-I3, L89). As a

side effect, “there was also some feedback on things that we can learn” (L91) as the asset team. Client community calls are a great business development opportunity, especially as each client has a different configuration (C2-I3).

Interviewee C2-I3 described how she presents to *associations* but mentions that they are protective about consultants coming.

Interviewee C2-I4 described a client representative who won an AI-related public *award* because of her involvement in the reuse of the asset featured in Case 2. External publications of this award indirectly helped in further marketing the reuse of the asset. The asset has won various internal *software awards*, to date, but no external award so far (C2-I3).

Interviewee C2-I3 is active on social media like LinkedIn and Twitter and plans to promote the asset via these channels. However, she has not done this yet. There is a *wide spectrum of marketing activities*. This includes international fairs and trade shows as well as webinars to potential clients. All these marketing tools are expensive and/or take a lot of time. Little is known about the effect these marketing efforts ultimately have on software asset reuse.

The team has decided against publishing an asset *newsletter* to be regularly sent out to clients, perceiving newsletters as a bit old fashioned (C2-I3).

References

The marketing of the asset depends on the available references. This asset has been licensed in big contracts to only a few clients. So, other than a product, the software asset has only a small client set which limits the number of references in the market. There is *only one good reference* the team can officially refer to (C2-I3). However, due to close client relationships, they have clients they can send new prospects to on an individual basis; it “is not a formal reference, but the CIO is really willing to speak to other people. So, he’s very open ... chatting with his peers in other organisations” (C2-I3, L28).

The team makes active use of its reference. “Whenever we can, we drop it wherever we can, because it’s building trust and *we are not very well known* ... for our expertise in this area” (C2-I3, L34). As the sale progresses, the client moves from paying more attention to references than to uses cases (C2-I3).

4.3.4 Case 2 Asset Reuse

Figure 4.14 presents the asset reuse topics that were raised by the interviewees in Case 2. These will be discussed now.

Discussion topics	Case 2
Asset reuse	
Asset benefits	...
Asset reuse	...
Systematic reuse	...
Drawbacks of the reuse	...

Figure 4.14 Discussion topics of Section 4.3.4 (source: author)

Asset Benefits

Assets provide a *competitive advantage* in the marketplace in the following three areas: service, technology, and solution configuration (C2-I3). This way a reusable software asset outperforms a one-off-solution.

In this client partnership, the client only pays *25% of the development cost* (C2-I4). A software asset is “a great *revenue generator*” (C2-I3). The asset “really helps ... sell a lot of other stuff too” (C2-I1). These are further benefits of reusing software assets.

Interviewee (C2-I3) reported of her clients, “Thanks to our technology they have reached lots of savings”. She described how the clients see the platform the software asset is a major component of as a significant driver of internal efficiency.

Initially, the asset was to be used for different company-internal clients only. However, C2-I1 pointed out that “a lot of the requirements we got from some of our commercial stakeholders, they’ve had some really good ideas that have been well received” company-internally as well. So, the IT provider *is internally benefitting from asset ideas that arrived in the form of external client requirements*.

Asset Reuse

A seller working closely with the asset owning team is driving reuse. Interviewee C2-I3 *attributes the three external asset reuses to one particular seller who has a deep insight into the asset and can explain it well to his clients*. All the external clients to date are banking clients, indicating a vertical reuse (Anasuodei & Ojekudo, 2021; Jalender et al.,

2010) as described in Section 2.5.5. It is a cross-industry software asset, so horizontal reuse is untapped yet.

For the SaaS asset in reuse Case 2, the number of users on the client side is increasing. This success is limited and reaches a plateau when the client's full organisation is on the platform (C2-I3). At this point, the reuse of the software asset at this client would have been exploited to the maximum.

Systematic Reuse

In 2017, the asset team contacted some *business* teams and realised “someone might be interested in using the same ... application because basically, every corporation has similar issues” (C2-I2, L105). This can be seen as some abstraction as discussed for the reuse of Case 1 in Section 4.2.4. Further, there was no solution in the market addressing “the big picture like whole corporate picture” (C2-I2). One executive “noticed that ... we have resolved an issue, which is troublesome also for external clients. So, he started to think. Okay, why not try to sell this product to someone else because they have some problems so we can give them solution to this problem?” (C2-I2). Subsequently, the asset team faced the new challenge of “think how to make the application running or ... box solution that for every client, we will set up an isolated instance, a new database, a new server application ... and so on” (C2-I2). This is how the software asset started to be reused externally.

The first commercial reuse of Asset 2 was a *horizontal* reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010) as described in Section 2.5.5 – outside the original domain where it had been reused in two separate instances (C2-I2, C2-I1). Overall, the asset has three major sets of clients – internal, Corporate Social Responsibility, and commercial (C2-I1). The first and original client set was internal – in various parts of the organisation. There are two application instances for Corporate Social Responsibility and one for every external client (C2-I1). Interviewee C2-I1 sees plenty more internal and commercial clients for the software asset.

Systematic reuse was described as standardising the documentation and solution to make the software asset more of a plug-and-play while offering additional service packages. This applies not just to the asset but to the whole platform ecosystem. “Then the reusability will increase significantly, and it makes it easier to sell” (C2-I3, L88).

C2-I1 (L427-L435) described the systematic setup of the asset for a new client. “Because we’re SaaS, systematic reuse is the name of the game ... we are a multi-tenant SaaS app ... if we ... get another customer, we spin up a new instance of [the asset] and we actually have a squad dedicated to systematic reuse. So, we call them the tenant operation squad. And what they do is they spin up a new tenant, and there’s some standard things that you have to do for every new tenant. You have to set them up with ... what we call a single sign on ... So, we set up assets ... at the enterprise level ... Pull in person data, fill in the trusted source connector data ... that team really is focused on that kind of systematic reuse.” This answer resonates with the Case 1 interviewees saying that systematic reuse requires a standard approach.

Interviewee C2-I4 described how he expanded the asset to make it fit his client when asked about systematic reuse. This was required because each asset was initially created for and tailored to the first client. As more clients come on board, the asset's functionality will change. It takes responsible thinking to decide what changes to include in the asset and what not. “You need to evolve the solution to *make sure that a maximum number of users ... can get benefits ... from the assets.*” (C2-I4, L119). So, every reuse includes the decision on new asset features and subsequently turns the steering wheel on the road of reuse. Adapting, e.g., for the cloud, turns the asset onto the highway of reuse. Because every reuse affects the asset, every decision for reuse is a strategic decision resulting from knowledge management (Wijnhoven, 2008).

Drawbacks of the Reuse

Interviewee C2-I4 talked about the drawbacks of software asset reuse. The development *cost for even minor changes is considerably high* because, in this cloud setup with just one code base, the effect of the change on the entire client set in terms of usability and performance needs to be analysed before this change can be implemented. Another disadvantage is the *prioritisation of changes*. Often changes affecting large accounts receive the highest priority (C2-I4). The software asset reuse ties the reusers to a virtual community that interacts directly or indirectly because changes to the asset typically affect more than one client.

Figure 4.15 shows the timeline of the reuse processes for Case 2 described in Section 4.3.

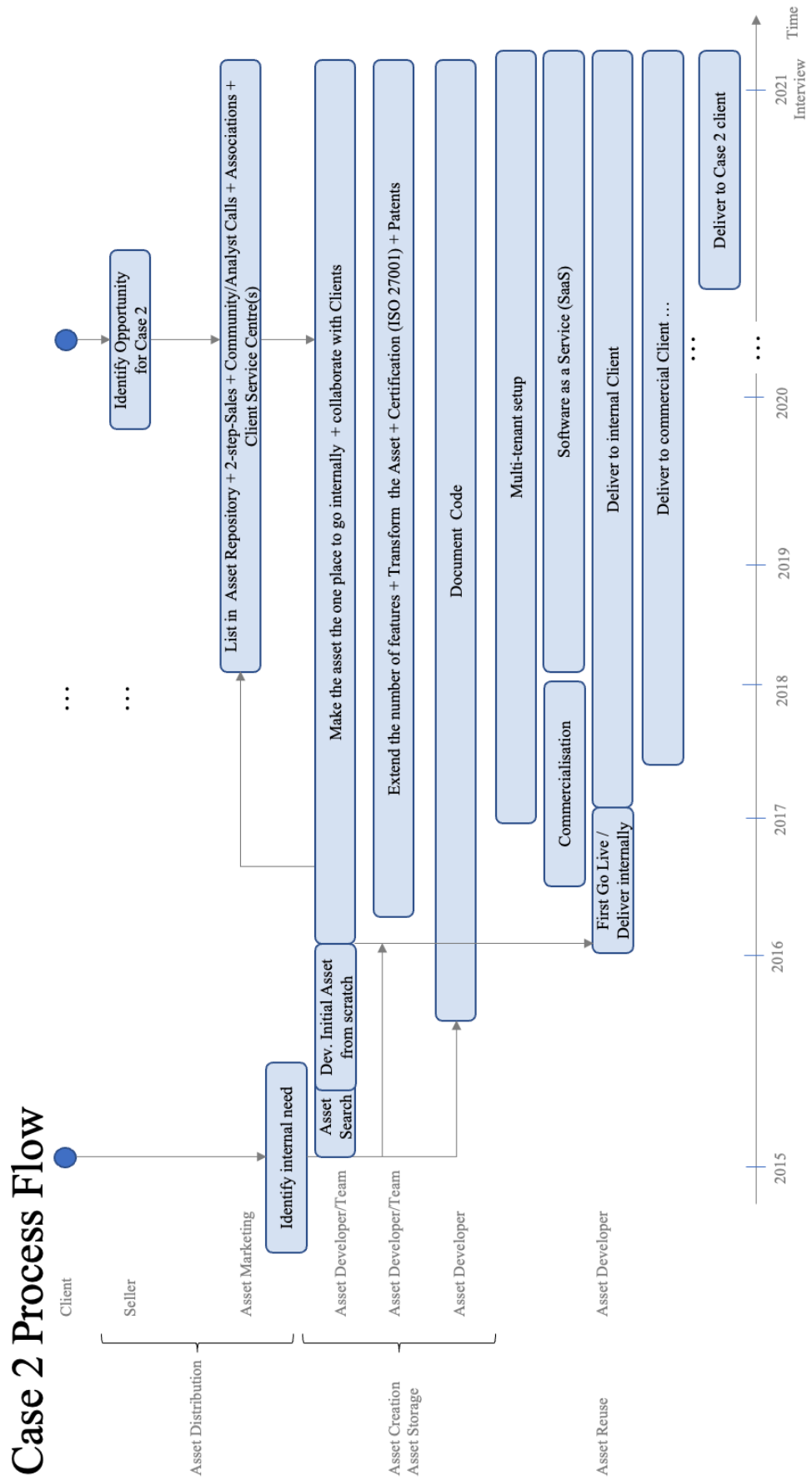


Figure 4.15 Reuse and process timeline for reuse Case 2 (source: author)

4.4 Findings Case 3 – Reuse Initiated during Solutioning Generating Revenue

The asset featured in Case 3 emerged between 2010 and 2012 based on key contributions from US, Czech, and Australian experts – see Figure 4.16. Circa 50 clients are reusing the asset (C3-I2).

Case 3 presents recent reuse at a Baltic bank as part of a deal focusing on infrastructure and application outsourcing, transformation and migration to the cloud. Reuse of the asset was decided on during the solutioning phase by the solutioning team and generates revenue in the form of a service fee. The asset is reused by a dedicated group of software testers of the IT provider and the bank working on the agreed IT deal.



Figure 4.16 Case 3 - Location of asset owners versus asset reuser - adapted from (Geology.com, 2021)

4.4.1 Case 3 Asset Creation

Figure 4.17 presents the asset creation topics that the interviewees raised in Case 3. These will be discussed next.

Discussion topics	Case 3
Asset creation	
The asset itself	...
Competition	...
Team	...

Figure 4.17 Discussion topics of Section 4.4.1 (source: author)

The Asset Itself

The asset was *brainstormed by a group of experts* consisting of two executives, the IT provider's research team, the India development team and the interviewee. They worked on the topic both onsite and virtually from distributed locations (C3-I1).

The team wanted to create a unique service that was unavailable in the marketplace at this time. The IT provider's thought leaders and internal research experts suggested a significantly different *proactive approach*. The objective was to create *one* big testing platform across the IT provider offering real productivity gains and greater efficacy (C3-I1).

Before creating the new asset, the team *checked what assets already existed*. They realised that about six or seven assets that made up the testing family were reused continuously. The asset team was using and engaging with these assets on a component-level basis. Beyond this, they were also searching for other similar assets (C3-I1). In contrast to Case 1, the asset in Case 3 was not created from scratch but purposefully shaped by reusing an existing set of assets (see (2) in Figure 2.17) (Mateen et al., 2017).

Interviewee C3-I1 had been searching for testing assets in this area for 20 years already; new assets are always emerging. There is no specific trigger that kicked off his search. His motivation was the permanent desire not to duplicate work and to be aware of what is already available internally and externally. What led to the final decision to create the new asset was to *stop the "never ending parade of unfunded assets* being developed within accounts that would partially replicate the functionality of what we were doing, maybe up to 25% ... which constantly frustrated us, because we wanted to ... pull them together into a combined effort to develop one single platform" (C3-I1, L35-L36). So, the objective of developing the asset was to be more efficient by bundling effort and expertise to build a comprehensive joint solution. This aligns with the reuse benefits listed in Section 1.3 (Barros-Justo et al., 2018). At the same time, an additional and powerful reason to create a reusable software asset was identified.

Competition

Asset owners respond to emerging competing software assets. As competitive tools emerge in-house, the asset team integrates these capabilities. If customers have external

tools such as Tosca, the Asset team integrates these as well as Salesforce tests, for example (C3-I2). Partnering creates a win-win situation for all competitors involved.

Team

The asset owning team consists of around 35 – 39 experts. Team members are the architect, business analysts, and developers. Key people such as the sponsor and interviewees C3-I1 and C3-I2 have been part of the asset team for a long time. The attrition rates are low. Most team members joined when the asset was created. As in Cases 1 and 2, the asset team is stable.

The global test offering leader sponsors this asset (C3-I2, C3-I3). He is “super charged, ... he's very much involved in ... discussing all kind of enhancements guiding us. What is the direction what our clients are looking for? So, I think that ... we get that energy from him” (C3-I2, L343). He is consulted and asked for his consent before implementing new asset features, as this impacts asset implementation and the asset-owning team's skill set (C3-I3). The sponsor is seen by the asset owner as the central owner of the asset and listed as such in the asset repository. He recently helped to reduce the cost of the asset, which helped the reuse team build an attractive business case for the client (C3-I4).

Asset 3 has *regional sales teams*. These teams demonstrate the asset to the client while the solutioning team helps to include the asset in testing deals (C3-I2).

Asset owning teams are self-funding. “There was not, ... at the time, sufficient interest to invest to the levels that will be required for product ... We developed it with *incremental funding*. And in that approach, we only had the opportunity to create an asset and not a product” (C3-I1 L5-6). The self-funding approach ensures that only successful assets can move on.

4.4.2 Case 3 Asset Coding and Storage

Figure 4.18 presents the coding and storage topics raised by the interviewees in Case 3. These will be discussed now.

Discussion topics	Case 3
Asset coding and storage	
Development	...
Documentation	...
Releases, versions, variants	...
Knowledge	...
Forgetting knowledge	...
Innovation	...
AI and automation	...
Patent	...
Cloud	...

Figure 4.18 Discussion topics of Section 4.4.2 (source: author)

Development

Only 25% of the code for the asset in Case 3 had to be newly created. The *main components of the asset had been built earlier* by the IT provider’s research team (C3-I1). This contrasts with Case 2, which was fully created from scratch. Once investment was received, the development started in 2010 and took approximately 2–3 years. The asset was developed in the first year. Then functionality was intensively supplemented, and maintenance resumed over the next two years. Adding new functions and maintaining the asset is ongoing. It is an “*incremental development, ... harvesting process*” (C3-I1, L68). The asset team constantly looks for new features to be harvested from account implementations.

This asset has no refinements for any industry or technology (C3-I3). The asset has been kept as generic as possible and will only be customised for a specific client.

Much innovation is client-driven, i.e., the client comes up with new requirements. Sometimes this only requires a simple API implementation. In other cases, entirely new features must be built. “They see a challenge and they come up with an idea or a solution to it” (C3-I3, L266). If the new feature created for one account could be relevant to others, it needs to be harvested and shared by the asset team. Additionally, an innovation the asset team has developed is called “Preview”, which allows customers to easily view the test scripts created at a glance (C3-I3). Client-driven involvement has not been found in

the literature. It is typical for reusable software assets where a small set of long-term clients exists.

Documentation

The asset is *documented* in the form of tutorials, user guides, training, and interlock training based on the IT provider's methodology to a level similar to that of a product (C3-I1). The asset team worked hard to make the installation and storage of the *asset code* as simple and seamless as possible. It is stored as raw and zip files, which can be easily passed on via ftp link for installation (C3-I2). A software asset comprises more than pure code. This confirms Bauer's (2016) list of reusable software artefacts.

Releases, Versions and Variants

The asset follows a development plan agreed with the executive and development management (C3-I1). Initially, the asset started with circa ten features, quickly rising to over 100 features (C3-I1). Any new features are shared in the form of new releases every month or two with stakeholders and account teams so that all are informed and can jointly benefit from them (C3-I1).

Initially, different components were sold as various accelerators as innovation. However, about 2.5 years ago, the asset team realised, following client discussion, that most are looking for an end-to-end solution. That was when the team decided *to pack all these different features into one solution*. This required much re-invention, e.g., in the form of AI and a machine learning-based automation engine. The main objective was to deliver that experience as a one-stop portal for any client testing need (C3-I2).

The asset comes in *three variants*: Windows on-premises, on the cloud and Linux. The asset can then be operated by the IT provider or the client (C3-I2). This adds complexity as there is more than one asset variant to be maintained.

Reusable software assets come in different versions. Over time, *circa ten versions* of the asset had been created. The previous desktop version was closed in 2019, and the asset owners now use the current version. Now, they are on version 6.3 (C3-I2). All versions will be rolled out for all variants.

The software asset has been created always to be *reused* unless the client runs an ancient legacy system or a particular technology the asset owning team has not dealt with before

(C3-I2). Being open to many different technologies across versions and variants adds to code complexity, which must be maintained over time.

This effort ensures that the asset universally applies to almost all testing opportunities. Even if the project has a very short duration, the team implements the asset as a value proposition. The only exception would be SAP automation, handled without the asset (C3-I3). *About 90% of the testing opportunities can be addressed with the asset.* In all other cases, there is a technical dependency preventing the reuse of the asset. Interviewee C3-I3 expects this gap to be addressed over time.

Different testing teams use the software asset. It is both an asset and an evolving *offering* with many different capabilities. It can be used for, e.g., Salesforce, AI, or performance testing. The software asset can be brought to the client either as a platform asset or as smaller, separate components, e.g., an automation module, a cognitive module (C3-I3).

The asset owning team continuously works to improve the asset and make it more robust and stable, resulting in updates every month or two. These updates are typically passed on to the existing clients so they can upgrade and benefit from the change. Updates are also communicated via the company-internal wiki page related to the asset (C3-I2). All these updates are reflected in each variant and the latest version.

Suppose an account already has a testing software platform in place. In that case, the account team tries “to do the pilot and ... look at ... what is the competitive advantage and the kind of value proposition we can bring in ... And then ... we also have our own lessons learned to take it back to what we need to enhance” the asset (C3-I2 L294). This analysis can result in new software asset features being created.

Knowledge

The asset represents different kinds of knowledge. “There’s a number of different levels of knowledge contained within the asset ... The first, is that it’s an expression of a unique, ... method approach towards quality engineering ... At another level ... contains ... cognitive models that reflect client experience in regard to defect and test case learnings ... Other knowledge is in regard to the [name of another AI-related asset owned by the IT provider] component, ... which is interlocked into it. So, there is a chatbot component, ... and obviously, there’s ... the ... coding elements coming together to provide ... the core functionality around the tool” (C3-I1, L118-120). The software asset presented in

Case 3 contains multiple generations of testing knowledge that evolved from different applications (C3-I2). This finding aligns with Section 2.4: software represents knowledge (Di Cosmo & Zacchiroli, 2017; Dreyer & Wynn, 2016; Huang, 2019; Kneuper, 2002).

The IT provider's staff who demonstrate deep hands-on knowledge of the asset can achieve a badge or certification, which they can share internally or externally, e.g., on LinkedIn. "We have had success in at least two accounts where we have done this, where the clients are very impressed, and they said, why can't you give a badge to us as well? So, they were very happy when they were able to get some acknowledgement having worked on it" (C3-I3, L407-408). This virtual badge is seen as a skill acknowledgement.

The asset owning team maintains knowledge. The attrition rate of the team is low. If a member of the asset owning team leaves, much knowledge is preserved because the team records many knowledge transitions. Further, extensive asset and knowledge documentation is available in the wiki (C3-I2).

To gather the knowledge to create the content and cognitive model of the asset, 150,000 defects were harvested from multiple client sites, analysed, classified and categorised (C3-I1). The chatbot was developed from existing test documentation and methodologies. The existing methods infused into the asset were based on the original test methodology written back in 2006–2008. A complete life cycle of testing methodologies later evolved into a more complex, end-to-end testing methodology cycle. Here a hybrid approach was applied: agile development, advanced approaches to testing, blockchain, and recent quality engineering approaches (C3-I1). Many evolutionary innovations added value and built up revolutionary innovation in testing. This confirms the details in Section 2.3 (Hargadon, 2002; Wegener, 2016).

Forgetting Knowledge

Some accounts have developed testing tool features that the asset owning team consciously decided not to take on board the asset. In this case, they skipped the knowledge and did not document it (C3-I1).

However, when it came to features the asset owning team developed themselves, the team made every effort to preserve these ideas. Interviewee C3-I1 explained, “my job was to be the rememberer of the organisation ... I just kept everything on C drive. So, I sort of became the ... the memory board for the [asset name] community platform” (C3-I1, L159-160). The old asset originally developed is “100% there, so ... the driving engine of it, ... which ... makes up an important 40 to 50% of it ... the core functionality is still there (C3-I1, L241-243). In contrast to Section 2.4.5 (Spender, 2008), asset-related knowledge is carefully preserved and not forgotten.

Innovation

A lot of innovation happens due to clients asking for improvements following technology changes or because there is a need to integrate commercial tools to be competitive in the market. Then the asset owning team tries to respond. Each of the approximately 70 current clients of this asset has a unique requirement and uses a combination of commercial tools to integrate with the asset. This triggers lots of innovative enhancement requests (C3-I2).

AI and cloud are future software asset-related trends (C3-I3, C3-I4). Further trends include agile, microservices, containerisation, security, confidentiality and data privacy (C3-I2). Currently, the IT provider has a long cycle before release for testing new functions and applications. The plan is to reduce this time significantly to speed up testing and deployment (C3-I4). This requires testing to happen faster and more frequently, which is new and innovative for a software asset (C3-I4).

AI and Automation

New features are constantly created. After doing some Proofs of Concept (PoCs), the software asset has recently been enriched with AI to become more powerful in test requirement analysis and implementation. Based on this automation, the software asset can create test plans, write test scripts and execute test cases automatically (C3-I2). This

new feature has not yet been promoted thoroughly. Promoting it will increase the asset's reusability.

Automation helps clients innovate faster. In another example, the team could *automate* about 70% of the test cases without having to write a single script (C3-I2). Because the Selenium test framework used can automatically generate scripts that only have to be executed. The constant improvement of the asset helps the asset to become more and more powerful.

Patent

There is no strict guidance on patenting software assets. Interviewee C3-I1 regrets having abandoned raising the patenting process for the key features. "Because we had a unique asset, and I should have somehow got the recognition for team and myself, the new unique aspects of it" (C3-I1, L73). More experience is required to understand the impact patents could have on the reusability of assets.

There are plans for several patents to be issued on the AI and automation part of the asset (C3-I2). This is because the asset requires consistent funding and support to continue with the research and evaluation of the asset. A patent will give the support to make the asset a more robust and commercially evolving solution (C3-I2). While the research team has developed an early prototype and completed it, the testing practice team is now working on an ongoing basis to turn the asset into a scalable and commercialised model (C3-I2).

Cloud

The asset started by being installed on the client side. The entire asset platform is cloud-enabled (C3-I1, C3-I2, C3-I3). While the asset code and the whole application reside on an Open Shift server, servers inside the IT provider's premises are being used. Further, there is a GitHub repository where the application data reside. The asset can be customised for AWS, Google or Microsoft cloud (C3-I2, C3-I3) if required.

4.4.3 Case 3 Asset Distribution

Figure 4.19 presents the asset distribution topics that the interviewees raised in Case 3. These will be discussed now.

Discussion topics	Case 3
Asset distribution	
Use cases and references	...
Asset marketing	...
Internal marketing	...
External marketing	...

Figure 4.19 Discussion topics of Section 4.4.3 (source: author)

Use Cases and References

Use cases are helpful during client presentations to illustrate the added value of the asset when it comes to convincing a client and winning a PoC or a fully-fledged implementation (C3-I3).

Over time, the *use cases evolved* from a user-interface automation framework to a fully-fledged platform on the cloud allowing continuous integration, with DevOps concepts for different types of testing, such as data and performance testing. It integrates with a broad spectrum of internal and external tools (C3-I3). The asset should not be marketed as a simple tool but as a specific automation platform which enables quality (C3-I1).

Asset Marketing

The interviewed reuser C3-I4 first learned about the asset circa five years ago when he received internal newsletters and invitations to internal education sessions for sales enablement on this asset. The asset is “not the same anymore as initially, so we did a demo for the client as well, and [it] was quite impressive how it has evolved” (C3-I4). It is important to share the asset owning team’s knowledge on newly implemented features with the account team, who can materialise the new asset enhancements (C3-I3). Despite all the efforts, it remained unclear if the latest asset capabilities had been fully communicated to the asset solutioning and sales team at the time of the interview (C3-I4).

The asset has a record in the asset repository providing some high-level information on the asset, including contact names. The asset's demo user guides, troubleshooting and installation guidance, best practices, templates, estimations, or case studies are stored on the asset's wiki pages (C3-I2, C3-I3).

Although the information on the asset is published and accessible to everyone, *employees repeatedly contact interviewee C3-I1 to ask him instead of looking up the information themselves*. “I find that people have been incentivised in some ways to artificially populate the knowledge repositories with information that really isn't that useful ... So, that there's a lot of murky information in there, ... and I'll constantly see people reposting ... presentations or products or IP that I've developed and saying that they're the author ... I stopped worrying about that a long time ago because it just wasn't worth chasing it” (C3-I1, L177-181). This confirms Terjesen's (2003) earlier findings on interpersonal knowledge transfer explained in Section 2.4.6: people prefer talking to people instead of anonymously reading repository content. It also confirms Oppenheim's (2003) theory on knowledge transfer, detailed in Section 2.4.2. Explicit, written knowledge is insufficient for potential reusers to understand the purpose and function of the reusable software asset. It misses out on the tacit knowledge belonging to asset owning experts.

Interviewee C3-I1 emphasised the importance of *selecting a suitable asset name*. It should not cause distraction or confusion (C3-I1). To achieve asset reuse, it is helpful if the asset and the offering are closely related (C3-I1).

The asset owning team hosts monthly *asset awareness sessions* and quarterly release-based sessions where the asset owning team tells people the new features. Further there is a fortnightly mail communication to educate staff about the asset, new success stories and their contacts. This all helps to make teams aware. “This is more of a push communication where we make sure that it rings a bell for people repeatedly. So, they go and cascade this information to the clients as well” (C3-I3, L104). Here the asset owning team reaches out to industry experts to service line or technology experts, e.g., Salesforce, Adobe (C3-I3).

Because this asset has *sales teams in different locations*, the asset owning team can go straight to market with the asset and engage with clients directly to present the asset's value proposition. Of course, “it's useful if there is an account team existing, and we do

have a relationship with them, it becomes easier to connect with them to talk about our ecosystem” (C3-I3, L234).

The global sales leader responsible for this asset scans the opportunity pipeline in the system to identify opportunities which can be addressed with the asset (C3-I3). This makes sense because this asset is more a methodology asset than a content asset. The Case 3 software asset allows certain testing methodologies to be applied. Asset 3 does not provide special core content to the client. In contrast to Assets 1 and 2, it is not the reason why a contract is being closed. It is limited to testing and acts more like an accelerator.

At the time of the interview, the asset had been *reused on about 50 client accounts* (C3-I3) and applied in contracts the IT provider has just started and extended contracts. The asset owning team has *some referenceable clients* they showcase during their client presentations, particularly when the asset owners can showcase previous reuse successes in the same industry (C3-I3).

Interviewee C3-I3 stated that winning asset-related awards has a marketing effect. The asset has won various external *awards*. It has also scored highly in the Net Promoter Score (C3-I3).

Internal Marketing

“*Everybody in the testing environment ... is expected to have the fundamental knowledge about this particular asset because all of our solutions today are using a recommendation about this tool. So, we go to the client and pitch in and say, what is the value proposition of using this asset as part of their delivery*” (C3-I3, L10-11). The software asset is part of the IT provider’s internal professional training. Interviewee C3-I3, the service area leader for building the capability of the asset, runs *training programmes* to educate the employees of the IT provider about the asset. “The more and more people are going to be skilled on this, it’s going to make our delivery much more easier” (C3-I3, L48).

Internal marketing is critical. The asset owning team is engaged in community work. They run a company-internal *chat channel* with nearly 5,000 subscribers who can scroll past discussion chats for knowledge updates (C3-I3). The asset owning team shares information on their asset in *regular newsletters* on the asset wiki (C3-I3).

The asset team can work in *agile* mode by using simplified pricing models (C3-I3).

New marketing material is constantly emerging because the new features, presentations, videos or partnership marketing material on wikis ages quickly (C3-I3). It is challenging for experts outside the asset owning team to stay current.

The interviewed reuser C3-I4 was keen to be informed about the *future roadmap of the asset*, e.g., when the next feature release is planned. This kind of information could be beneficial to share with the client. While the client did not ask for this information, it would have been interesting and relevant for future opportunities.

External Marketing

The asset is regularly presented at *analyst forums* like Forester or Gartner, to make the testing industry aware of the asset (C3-I3). The asset is demonstrated in *Client Briefing Centres* for clients to come and see it live in action (C3-I3).

Partnerships can expand the reuse area for software assets. The asset owning team closely supports the sellers or third-party partners in jointly bringing the asset to the client (C3-I3). Going to the market with third-party teams requires the asset reuse teams to listen live to their way of supporting asset reuse. Typical third-party providers are Salesforce, SAP, ServiceNow and others (C3-I3). “We are working together with them to get into a lot of different areas. ... I think that's a great win ... for us. That's helping us reach across many other clients as well who would not have typically come in if we had work silos.” (C3-I3, L449-451).

Prior to closing the final contract on the reuse of the asset, the client team watched a demo on the asset and ran a pilot (C3-I4).

4.4.4 Case 3 Asset Reuse

Figure 4.20 presents the asset reuse topics that the interviewees raised in Case 3. These will be discussed now.

Discussion topics	Case 3
Asset reuse	
Reuse benefits	...
Reuse drawbacks	...
Asset reuse	...
Support	...
Systematic asset reuse	...

Figure 4.20 Discussion topics of Section 4.4.4 (source: author)

Reuse Benefits

Over time, the *revenue generated* by the asset is continuously increasing. More and more accounts became involved in adopting the asset (C3-I3). The longer the asset exists and the more intensively it is being reused, the bigger revenue will be generated.

The *savings* were identified during the sales cycle and considered in the business case “if we have efficiencies through the asset over time to save some manual work” (C3-I4, L252). The savings are typically hard to measure and therefore, not tracked.

The asset helped in the client conversation. “The cool thing was that in the conversations with a client around testing, and the issues that they had previously, we could always refer back to the asset and say that we do have functionality or capability within the asset that can address those problems. So, we always did *have an answer for their pain points.*” (C3-I4, L277-278). The software asset allows the IT provider to step in as the expert.

The asset “gives us *a thought leader position* in the overall testing industry itself ... because of the approach ... It shows to the industry what we have as a holistic ... team, rather than just being body shoppers. ... This tool helps us penetrate into different areas, because we are having an integration with multiple other components and with multiple other technologies as well. I think that's the win-win” (C3-I3, L173-175). This aligns with the previously discussed cases – reusable software assets help providers compete in the market.

Interviewee C3-I4 felt motivated to include the asset in the deal with the Baltic bank “To get the deal signed, it was a *differentiator* for us ... because the [asset name] is a platform of open source software and the client has a strategy of open source ... but it was a combination... the client would have not chosen the ... asset, they would have ... had to *build their own platform*, ... which is a bit more complex and also harder to maintain or

they would have had to go for a product off the shelf, which is in more cost in terms of licences which is commercially not so viable. So, it was also a commercial aspect ... or benefit.” (C3-I4, L24-27). The client recognised the commercial benefit of software asset reuse.

Interviewee C3-I3 reported the feeling of *reuse success*. “Whenever we start a project, there is always a scepticism that is there with the clients because it's a new tool and the team is also pretty new. So, all of us and many new projects, we run into a rough ... weather. Initially, the team is struggling, and the clients also are very keen to look at what we are going [to] deliver [to] them. But then, at the end of the release, when they see a great productivity benefit coming out of this whole story, I think, uh, that's the happiness we are able to derive by implementing this whole thing.” (C3-I3, L431-432). Reusing a software asset is not an easy decision; it is an investment in the future.

Reuse Drawbacks

There are *not many statistical success metrics for reuse*. One is how many clients are using the asset. Another alternative is to measure client satisfaction via NPS (C3-I2). While these can be precisely determined, these two metrics do not indicate the financial benefit.

The financial benefit is hard to determine, which may be one reason *why no specific reuse target* had been assigned to the team. Interviewee C3-I3 reported having a target for building the asset capability. This is measured by the number of people skilled in this area, i.e., how many people are in the particular job role as “test specialist”. “We have under this category ..., typically every quarter or every year, ... some targets to, ... win more accounts in this area. ... We cannot have somebody directly moving in this ... niche. ... We cannot hire people in this area. So, all my people will be coming either by internal mobilisation or by internal reskilling” (C3-I3, L62-64).

The asset is not up for independent sale. Instead, the IT provider’s account *delivers services to the client based on the tool*. The asset owning team supports this account team, e.g., by developing and debugging the software asset (C3-I2).

It was unclear to any member of the asset owning team if and where the reuse of the asset is being tracked. “We are able to capture *how many people are requesting for access* to it to look at the models or look at the test scripts. So that way, we are able to see the number

of people doing it. And that gets reported to the offering leaders” (C3-I3, L210). The extended asset owning team may not receive feedback on their success by not making this information available.

Interviewee C3-I2 shared how the asset team was brought in during a cloud migration. Using the asset, they found that the existing test cases had only 35% coverage. By fully utilising the asset, they *increased the test coverage to 100%*. In addition, they reduced the test execution effort by 90% as the number of test cases came down (C3-I4).

Asset Reuse

The asset owning team is interested in becoming involved in a client opportunity early to identify code defects as soon as possible. However, even in the worst case, it would still be possible to bring this testing asset in much later, e.g., if the client is facing challenges (C3-I3). This testing asset typically acts more like a tool or accelerator.

The IT provider’s account for the Baltic bank client comprised, at the time of the interview, circa 20 people and will grow up to *50 people, including testers and test leads* with different focuses like regression testing, test automation, and functional testing. However, the IT provider’s testing experts will be testing only a subset of the applications as the client still does some testing in-house (C3-I4).

The asset management team does not always *charge for the assets*. It depends on how the customer will use the asset (C3-I3, C3-I4). If the asset is used solely by the IT vendor team, there is no cost. However, a licensing model is introduced when both the customer and the IT provider conduct the testing. The pricing structure depends on the number of people using the asset and the years the contract is for. If a customer cannot cover the asset fees, the IT provider can show certain flexibility in its pricing model in exceptional cases (C3-I3). The fee for the asset is included in the business service fee (C3-I4). This means the exact revenue generated by this reuse of software assets cannot be specified.

The past *reuse of the asset was “more of a push* from the practice site because ... they have been educating the different accounts ... and one account helped us go ahead and do it in ... multiple other accounts” (C3-I3, L143). This confirms what was discussed in Section 2.4.7: pushing software assets into reuse often makes sense instead of waiting for ad hoc reuse to happen eventually (Wallace, 2018).

The solutioning team tried to bring the asset into the RfP as early as possible. But initially, testing was not in scope during the nine-month sales cycle of the tender. The solutioning and sales team discussed with the client to include the testing scope. “It was good that we presented it from the beginning. Even if it was not in scope ... it wouldn't have been good if we only started that [reuse discussion] at the end” (C3-I4, L112).

Interviewee C3-I4 described how he submitted the asset in all five submission rounds of the tender. “Even if it's not in scope to bring it right from the beginning as an option ... you can at least have the discussion” (C3-I4, L116). This again reflects a push situation. Reuse had to be consciously driven to make it happen.

The solutioning manager for testing *proposed the asset for the deal* (C3-I4). The asset was first considered in September, proposed in December 2020 and then the client accepted. The solutioning was done in April 2021. The asset was suggested after it became apparent that the client did not have a tool. Interviewee C3-I4 was in charge of designing and selling the solution from a commercial point of view and positioning it towards the client. Further, a solutioning team worked on the technical details (C3-I4). Including the software asset within the opportunity was a multi-step process.

During the RfP-process, the client decided not on the asset individually but on the whole package. “The decision was done by several client stakeholders – C minus 1 level mostly. ... The executive sponsor is the CTO. So, he gave the final proof from my side, and then it had to be approved in the supervisory board.” (C3-I4, L63-64). The software asset was not in the centre of the RfP. Instead, it was seen as a tool to ease the project implementation.

The last moment to include the asset into the deal would have been “*when the client basically closes their budget*” (C3-I4, L119-120).

The client was out of support from his current testing tool and needed a new tool. He applied selection criteria when choosing the asset: it should be an open-source asset, and technical aspects like functional test case coverage should support test case automation for mobile and regression testing. The proposed asset was a good fit.

While selecting and including the asset in the deal went smoothly, there were issues with including the asset in a long-term deal. The available company-internal online calculator provided by the IT provider's global asset team allowed only a contract duration of three

years, while the given contract has a duration of five years (C3-I1). Reuse requires various administrative steps for all parties involved.

Support

The asset owning team offers an initial *three-month installation support* and then Subject Matter Expert support to make sure the asset setup runs stably. “For every client implementation ... we have a separate *SPOC* [Single Point of Contact] for all the delivery accounts. ... The responsibility of this SPOC is to interact on a weekly basis, or on a bi-weekly basis to understand ... what is the ... gap or what is the defects or what is the use case that is required for them to ... go ahead with their ... implementation so that would be sometimes a release where a particular mobile feature is missing. So, then we prioritise that as part of the development team’s work to ensure that it is delivered before the accounts release. So that they can reuse it and deliver it for the client” (C3-I3, L386-388). Pushing the software asset into reuse requires the leading person from the asset owning team to plan, agree, and implement across all the involved teams.

Systematic Asset Reuse

Fast responses matter to clients. “One of the key things to driving systematic reuse is ... *quick pricing* policy and the correct ... pricing approach ... We had to make sure ... that we priced it ... so that it would actually support ... the maintenance effort ongoing. But we also still ... had to make it cheap enough and easy enough to engage, such that it would be adopted without thinking within a pricing model. ... Needed to be able to create a pricing model that was easy to communicate, ... and also easy to implement into the pricing process. I think that was the major roadblock” (C3-I1, L246-250). The service fee for the asset reuse must be provided at least as quickly or faster than an estimation for new coding. Pricing policy, a crucial aspect of knowledge management (Wijnhoven, 2008), is applied.

Interviewee C3-I2 sees systematic asset reuse achieved because the asset owning team can bring the software asset to *more than 70 clients* without much customisation. Only a few minor enhancements or a little customisation are needed to configure the asset for the next client. It is “a very balanced combination of reusability as well as configurability to the client” (C3-I2, L246). The three asset versions, cloud, Windows and Linux, can be installed, and these can be used 90% of the time without any further configuration (C3-I2).

Systematic reuse is achieved by “keeping our sellers and solutioners aware of the importance and the *value proposition* of this asset. We try to ensure that this gets solutioned because this ... is enhancing the value proposition of the overall solutioning deal. So automatically, this gets included in almost every deal” (C3-I2, L264).

Systematic reuse can be achieved when *awareness programmes* are rolled out that educate the staff on the latest asset capabilities, if the teams understand how to use the asset and how to take the asset to the client. This awareness program includes various target groups. Dedicated asset enablement material is available for the sellers’ group (C3-I3).

Interviewee C3-I4 felt that systematic reuse has led to the asset reuse on his deal. “So, the *solutioning team* that is actually building the solutions from [the IT provider] side, they need to bring that forward practically, because not all sellers know all the ... assets” (C3-I4, L32-33).

Figure 4.21 shows the timeline of the reuse processes for Case 3 described in Section 4.4.

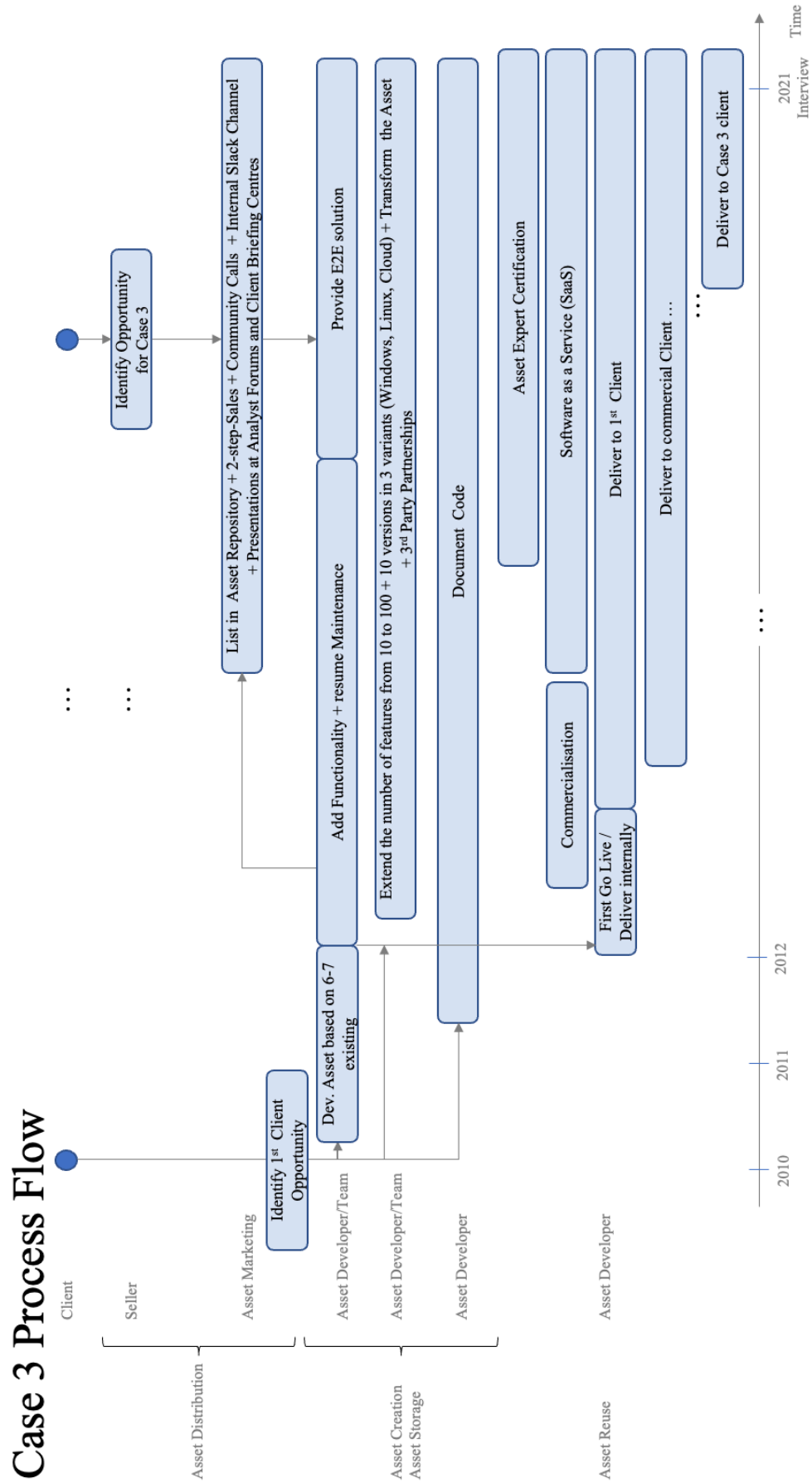


Figure 4.21 Reuse and process timeline for reuse Case 3 (source: author)

4.5 Findings Case 4 – Reuse Initiated during Solutioning Generating Savings

The Case 4 software asset is a platform created in 2012 for jumpstarting specific big transformational third-party software implementation projects. The asset creating and owning team is globally sourced but mainly in the US and Ireland – see Figure 4.22.

An industrial client in Sweden reused the software asset. This reuse was decided during the solutioning phase by the solutioning team. It generated savings both for the client and the IT provider.

The reuse case C4 was a situation where neither the client nor the IT provider project team had agile experience. On top of this, the pandemic forced 100% virtual working for the first time. Rolling out the asset with hundreds of people involved was a real challenge under these circumstances (C4-I3). This was the asset's inaugural use in this country, as it coincided with the first project tailored to its specifications. The client's and the IT provider's project teams had to be trained (C4-I4).



Figure 4.22 Case 4 - Location of asset owner versus asset reuser - adapted from (Geology.com, 2021)

4.5.1 Case 4 Asset Creation

Figure 4.23 presents the asset creation topics that the interviewees raised in Case 4. These will be discussed now.

Discussion topics	Case 4
Asset creation	
The asset itself	...
Competition	...
Asset team	...
Awards	...

Figure 4.23 Discussion topics of Section 4.5.1 (source: author)

The Asset Itself

Besides creating new software, the IT provider is actively rolling out standardised enterprise software packages like, e.g., Salesforce, SAP, Workday, Oracle Cloud HCM, PeopleSoft. Case 4 looks at a software asset created to improve efficiencies during the roll-out of one such third-party software package.

The focused software asset in Case 4 was created in 2012 in the US (C4-I1). There were three male and one female US-based delivery executives in North American and global roles who brainstormed the asset as a team of experts.

Software assets help improve efficiency. “A high degree of similarities and things that we need to accomplish on our programmes” motivated the creation of assets. “There's *a lot of things that we repeat* and that we recreate, ... The reason why I'm passionate about assets is ... because ... it brings a degree of consistency and quality cross projects ... it essentially reduces effort and [redundancy] ... and allows the team to focus on ... value-adds that ... differentiates the specific team on the ground and [name of IT provider] in general with our clients.” (C4-I3, L5-8).

Before this asset, since the 1990s, there had been a similar asset for an *earlier version of the third-party software package*. It was of much lower functionality and had fewer features. With this new asset, the concept evolved with a stronger foundation and broader functionalities (C4-I1). This can be seen as a software evolution, as mentioned in Section 2.5.2 described by Vizcaino (2017).

The asset itself is constantly adapted and updated. Following beta testing, the third-party software product provider issues a new version yearly. So, the clients who began to use it two years ago, are on a two-year-old platform and will continue with it until they decide to upgrade. Every *update* requires new funding. If the client chooses to stay on his old platform, additional new functionality will not be effective (C4-I1). However, interviewee C4-I4 *assumed that the asset will not be upgraded yearly* over the five years of the project because, in later phases, the asset will become less relevant.

Competition

There is no similar asset at the IT provider and *no internal competition*. However, the asset consists of various artefacts and tools for which sometimes certain alternatives would be available. Also, when the IT provider acquires new companies, they bring in new tools, too (C4-I1). These smaller artefacts may lead to improved ideas for the software asset.

At the time of creation, there was *no similar asset on the market*. However, over time, new employees joining from major competitors report having seen a similar investment at their previous employers. Interviewee C4-I1 assumes the functionality, breadth and depth of this offering to be much wider compared to the competition. This is because the assets built by competitors contain additional features on top of the core. In contrast, the asset featured in Case 4 provides key functionalities the customer needs (C4-I1).

Asset Team

Circa 126 members of the asset owning team are located in the US and in India. They drive global asset deployment in every region of the world where a third-party product is required (C4-I1).

Taking care of the knowledge contained within the asset, the asset team is split into *different streams*. Interviewee C4-I3 is part of the technical stream doing all the technical upgrades. Other experts form the functional teams or teams deal with finance, warehouse, business material, or process management (C4-I3). These work streams have acquired deep knowledge in their areas of expertise.

An *asset launch team* has been formally put in place two or three years ago. Before this, the asset launch was, after a formal hand-over, more a solution handover reliant on the onboarded project team. Now this handshake still happens, but the work is with an asset

launch specialist like interviewee C4-I3 who had undertaken many asset launches before. The role of the project launch team is to orchestrate all actions (C4-I3). Their job can be seen as equivalent to the software asset deployment experts in Cases 1–3.

Due to the complexity of the asset, the reusing team requires support. In the past, there was a system hosted by the IT provider with this asset. Now that they run the bigger project, they have a customer-owned system environment (C4-I4). When in need, the project team can ad hoc *contact two experts from the asset team*. The project team's questions are treated as high priority by the asset team, who respond very quickly (C4-I4). The reusing *project team* receives help from the asset owning team when installing the asset: there is a resource pool.

However, despite early support, interviewee C4-I4 suggested having *one dedicated resource in the reusing project focusing on methods and tools* who adapts the generic asset to the current client's needs. This expert should be responsible for making the software asset reuse a practical success.

The asset is sponsored at multiple layers, most effectively by the global service line leadership. However, the asset team also works at the industry level by providing a cross-industry solution (C4-I1).

Awards

The asset owning team is very motivated and passionate about their asset. Interviewee C4-I1 mentioned that the asset team received some *recognition*. The real motivation, however, was to see the corporation spending money on building differentiators for industrial customers (C4-I1).

4.5.2 Case 4 Asset Coding and Storage

Figure 4.24 presents the asset coding and storage topics that the interviewees raised in Case 4. These will be discussed now.

Discussion topics	Case 4
Asset coding and storage	
Documentation	...
Development work	...
Forgetting knowledge	...
Innovation	...
Cloud	...
Patents	...

Figure 4.24 Discussion topics of Section 4.5.2 (source: author)

Documentation

The asset is *well documented* and does not rely on an individual's memory (C4-2). The size of the asset is about 290 GB. This makes the asset too big to store in GitHub or in a virtual folder. Therefore, asset documentation is spread across different tools. Some asset documents are stored within the third-party product; some are in Microsoft Word or Excel and then stored on a cloud drive belonging to the IT provider (C4-I1). Some are kept via database backups and on hard disk (C4-I2). The documentation contains knowledge and is maintained by the asset owning team. All the initial documents used to create the very first version of this asset still exist (C4-I1). The externalisation of knowledge helps maintain knowledge, as explained in Section 2.4.3 (Dreyer & Wynn, 2016).

Development Work

The asset does *not use coding language* but represents a modification or refinement to the third-party product – a configuration (C4-I2, C4-I1). New features are constantly added due to technology changes, learning from past projects or scope enhancement.

Every quarter, the asset owning team *adds new features* to the asset. This is required because of changes in technology. It is an ongoing process, confirming the dynamic capability theory (Marrucci et al., 2022), necessary to stay competitive. The changes add new functionality, more features or redo a few things because of changes the third party provides to the code design (C4-I1).

If improvements result from the work in a specific project, this can be acknowledged by the asset owning team and *added to the general asset base* (C4-I2).

The asset is *continuously changing and expanding*. Interviewee C4-I1 explained that 14 months ago, the asset covered 220 cross-industry processes; now, it covers 600. The reason for almost tripling the number is that the third-party provider issued two new versions during these 14 months that affected 600 processes which required an update. Initially, the asset team concentrated only on the third-party software product. Now they have also integrated several external tools, including tools offered by the third-party software provider.

Forgetting Knowledge

Asked about *forgetting knowledge*, interviewee C4-I1 confirmed that there are certain features they do not have to remember because technology has changed. This is when the third-party software product provider stops supporting older versions after some time. It can be concluded that knowledge is only disposed of by the asset owners if no roll-back is possible; only then will Spender's (2008) knowledge forgetting apply.

Innovation

Kicking off reuse “works very, very well, but it doesn't work perfectly” (C4-I3, L110). To make it work perfectly, the team introduced the *agile work mode*. For “each project that we launch, we have a set of tools; we have a set of plans. We have ... templates ... in place. We have a launch plan in place and ... Our goal is ... to not only harvest assets, but we also do retrospective ... to understand what [went] well, what didn't go well, and we try to bake that in. So even our launch plan, which is part of eco plan, even that has evolved over the last few years based on ... lessons learned from previous launches” (C4-I3, L111-113). The asset owners closely monitor and reflect on their software asset reuse by applying an agile work methodology described in Section 2.5.2, which drives continuous improvement (Behutiye et al., 2022; Zorzetti et al., 2022).

Innovation comes with every new version of the third-party software product this software asset closely relates to. The asset team is also introducing new *innovative business processes*, automation, testing, cloud and agile working (C4-I2, C4-I3). Machine learning and intelligent workflows have been injected into the solution (C4-I3).

Automation supports the reuse of the software asset. The asset team can offer their clients automated scripts in Worksoft. This way, users can run *automated testing*. Currently, the asset team is working with other global teams to add features like robotics process

automation (RPA) that do not exist yet (C4-I1, C4-I3). The client could also benefit from cognitive automation via chatbots (C4-I4). All three main kinds of automation – RPA, AI, and testing – can be applied.

For example, provisioning a system manually typically takes almost two days. Using automation, e.g., via Red Hat Ansible, provisioning can now be done in three to four hours. The IT provider's automation team provided the playbooks required to set this up under the guidance of the asset owner team (C4-I2). Automation helps both the asset owning team and the client deliver software projects faster and more efficiently.

Cloud

The Case 4 reusable software asset has been brought into clouds like Azure, AWS, or Google. This allows the entire IT landscape to be fully provisioned, managed and maintained, which otherwise would have been done by another team for any on-premises installation (C4-I2).

The move to the cloud drastically *reduced the time to upload* the asset. In the past, splitting the file into smaller files took six to seven hours, plus an extra three to four hours. Now it takes two to six hours, and the files do not have to be split (C4-I2).

Using the cloud, the 500 plus process models can be uploaded (C4-I3). This *helps with storing the asset*.

Patents

There are no patents on this asset (C4-I2).

4.5.3 Case 4 Asset Distribution

Figure 4.25 presents the asset distribution topics raised by the interviewees in Case 4. These will be discussed now.

Discussion topics	Case 4
Asset distribution	
Internal marketing	...
External marketing	...
Mapping against offering	...

Figure 4.25 Discussion topics of Section 4.5.3 (source: author)

Internal Marketing

There are *quarterly internal enablement sessions* by region on the asset-related method, tools and accelerators. Additionally, the asset team runs a quarterly webinar series covering North America, Europe, the Middle East or India. Or, the team runs a three-week enabling academy session related to the third-party product, including teaching on the software asset. For all practitioners, many sessions organised by the asset owning team exist. They are recorded and publicised via the monthly global leadership newsletter. This way, the asset-related sessions receive much awareness and excellent participation in all the campaigns (C4-I3). The interviewed reuser of the asset, C4-I4, has attended asset education in the past. She reported that she will act as a teacher for a one-hour session multiple times later in the year (C4-I4).

The asset is embedded in the global service line communications. Practice leadership is involved at global and regional levels regarding enablement or client campaigns. This way, *internal marketing* is driven top-down. “There's such a good communication from top down with monthly communication coming up from our leadership on what are the assets, what ... are the enablement sessions coming up. But there's also a high degree of awareness on what our assets [are] within the practice” (C4-I3, L177). Interviewee C4-I4 confirms excellent communication on the method and the asset as such. That is why interviewee C4-I4 was familiar with the asset before reusing it.

There is a high degree of awareness of the asset within the practice resulting from executing *smart governance* via the monthly newsletter communication by the leadership, regular awareness sessions and a global bi-weekly interlock between the solution team and the asset team in each location. The asset owning team tracks all the deals where their asset has been proposed on all continents. Bigger deals get a certified solutioner assigned.

This way, the “deal automatically gets on our solutioning radar, and once it's on our solution radar, we are interlock[ed] with our solution team. So, ... we have ... access to that ... deal” (C4-I3, L194).

The asset team is not actively approaching *key sellers* and asking them to sell their asset because they are part of the practice and are already involved in selling the asset (C4-I3).

The asset is shown in physical *client centres* and demonstrated upon prior booking (C4-I3).

In about 50% of the cases, the asset team contacts an opportunity team. In the other 50%, the opportunity owner finds the asset team (C4-I3). Cases, when the opportunity owner finds the asset team, happen when the asset is not naturally a good fit or the opportunity, is below a pre-agreed size the asset team is focusing on so that the opportunity falls between the cracks of the agreed governance rules. In this event, typically, the opportunity-owning team knows the asset and contacts the asset team to obtain confirmation that the asset is a good choice. When asked for a fitment analysis, the asset owning team responds with an assessment that gives a clear answer (C4-I3).

External Marketing

Client references are important when responding to an RfP or RfI or live demonstrating the asset to a client. They allow differentiation from the competition and demonstrate expertise based on previous experience. If the client is not referenceable, the asset team will only mention the industry they reused the asset in. During client presentations, the asset owner always shares his personal asset reuse experience providing details on where and how the software asset has been reused (C4-I3).

To make clients aware and prompt invitations for high-level assessments, the asset team attends industry sessions, symposiums, gatherings, and asset-related conferences to demonstrate the software asset's differentiating capabilities (C4-I3).

Further, the asset team actively contacts *user groups* of the third-party software the asset is based upon (C4-I3). This way the asset team uses *external events* to propagate the benefits of their software asset.

Demonstrating the software asset to the client *helped win the Case 4 project*. The asset is an accelerator to the customer, a selling argument to justify selecting this IT provider.

The promoted asset gives the client confidence that the IT provider can do the job in a way that reduces the timeline needed for implementing the whole project (C4-I4).

The first client who reused the asset was an energy & utility *client who is still reusing the asset now* (C4-I1).

Mapping against Offering

There is a limited set of five to ten *offerings related to this asset*. That makes it easier to determine where the asset is a good fit (C4-I3).

4.5.4 Case 4 Asset Reuse

Figure 4.26 presents the asset reuse topics that the interviewees raised in Case 4. These will be discussed now.

Discussion topics	Case 4
Asset reuse	
Asset benefits	...
Asset reuse	...
Systematic reuse	...

Figure 4.26 Discussion topics of Section 4.5.4 (source: author)

Asset Benefits

Software asset reuse provides several *benefits*. The asset acts as a differentiator and thus helps to win more business. Further, it helps the team globally so that resources remain updated with the latest features and technologies; the asset team bundles skills that would otherwise be distributed across the corporation. Once experts are trained on the asset, they can do more effective work for the customers. So, customers get more value for their money (C4-I1).

The asset is said to *reduce the implementation effort* of the third-party software product by 30%. “We don't know yet. They have not come that far, but I think it is ... easy in theory, but not that easy when you work practically, and the one challenge is always about getting all resources up to right skill level that they know about it ... exactly the way you're working and ... it's as many people involved in such a project, and both the customer and the [IT provider] resources need to work in the same way. ... So, I'm not sure that this 30% gain will actually be true, but that's ... the way ... the project has been

estimated. And so, ... our timeline and our resource plan is reflected by ... our promise ...” (C4-I4, L82-85). Exact savings are difficult to determine.

The asset “process modelling tool has over ... 600 process flows ... if you were to ... create all of ... those assets from scratch on the project that's effort required ... So, we do charge the projects for the assets with the understanding that the *efficiencies gained from using [the] asset* will offset the cost and our solution team has some estimating tools that they can use to kind of provide that comparison to say look, if you don't use the ... assets, here's the effort you would need to create these on your project. And if you do use it, here's this, here's the reduction in effort” (C4-I3, L222-224).

“When we respond to ... our RfI's and when we go to orals, we're *differentiating ourselves from the rest of the competitors* with our assets and ... our method and our experience ... That plays together as a differentiator.” (C4-I3, L303-304). Interviewee C4-I4 confirmed that the asset acts as a differentiator in the market.

The asset team charges the project team for the reuse of the asset. According to guidelines, the asset's savings should offset the cost (C4-I3).

Asset Reuse

The reusers of Asset 4 are the IT provider and client teams (C4-I2). Reusers must pay to reuse the asset – before receiving the asset details. Typically, it is a fixed percentage depending on the estimate of the agreed software.

While the asset reuse was a commercial agreement, the client received the asset *free of charge* (C4-I4).

Two types of campaigns drive the reuse of the asset. First, there is a push model where the asset is offered during a *high-level assessment* of the current systems and processes. It results in a report based upon which the asset team then explains to the client how the asset can bring value due to its methodology and services. Alternatively, the asset team responds to an RfP or RfI, explaining the advantages if the asset is applied and the IT provider is selected (C4-I3).

The asset and its adjacent versions are significant in scope and quite complex. They are refreshed on an annual basis (C4-I3). There is a great demand for the asset. Reuse, therefore, occurs in two ways: getting the asset refreshed annually and getting the asset

to the programmes that need pre-sales or are delivering. There is *no asset reuse target* as such (C4-I3).

The *asset was identified very early* during PoC with this client. The client had worked with another competitor before but then contacted this IT provider and had a few meetings. The IT provider then “offered a small proof of concept project where we said, ... we will use this asset in a four months project to show you both our way of working, but also to model a simplified scenario within a [name of third-party product] system and then using ... this assets ... we run that project, and it was, in the end, successful so we could show them a demo of the system for their scenario. And then ... we won the big project that we are ... now running.” (C4-I4, L51-53).

Interviewee C4-I4 reported that the IT provider’s project team did not assess the asset in any specific way. “We assumed it would be applicable and ... we involved ... these experts ... to get us up to speed and to help us ... instruct ... all ... people ... in way, you're working and how to use it.” (C4-I4, L95, 99).

The asset reuser became aware of the asset during education. There was *internal communication* with links to a website as part of the weekly newsletters containing information on asset education (C4-I4). The reuser attended the training and will be hosting the training.

The strong need for the asset was demonstrated when the asset team updated their asset to the new version of the third-party software product. Then, one of their biggest clients contacted the asset team asking for knowledgeable people to help before the final release of the latest asset version. Eleven experts were identified and helped the client while the asset was updated in parallel (C4-I1).

Implementing the asset solution takes three to four months (C4-I2). Once the agreement is in place and the IT provider receives the order, the asset team will share the backend of the asset.

The project faced *practical issues* with the currency used in the asset. The software code was set to handle US dollars, but the project in Sweden required Swedish krona, which turned out to be a problem the asset owning team did not solve (C4-I4). The asset owning team missed out on customising the asset for the respective client.

The asset is ideally brought to the client at the pre-sales stage during RfI or RfP. Alternatively, it can be presented in *demos with the client's C-level executive* or other management whenever there is a hope that the client will move away from their old system to the new system within the next year or two. However, once a deal has been closed, it will be too late to bring the asset in since the approach, tools, assets, methods and teams have already been agreed. The governance ensures that clients normally know about the asset before signing their deal (C4-I3).

The asset team launches the programme once the opportunity team wins the deal. The programme team makes the asset available for use by the onshore opportunity team and keeps the asset updated. Every year, the asset is refreshed to be up to date (C4-I3).

During reuse, the asset stays with the IT provider team or their subcontractors, not the client team (C4-I3). However, when the project team leaves, they *leave the asset with the client*. By this time, the asset has been fully customised to this client. A master service agreement with the client ensures that the asset is shared with others without extra permission required from the IT provider. This applies both to implementation and maintenance (C4-I3).

Since the first asset of this kind was *introduced in the late 1990s*, the reuse has been increasing, but there have been fluctuations (C4-I3).

In bi-weekly calls, the *reuse of the asset is tracked*, and the status report is shared every month with the global leadership team in charge of this service line (C4-I3).

“Our goal is really to make sure that assets can ... have this *common denominator* ... that can fit the maximum amount of clients” (C4-I3, L360). Reusing the assets with many clients is easier when the clients are on a comparable innovation level. A special asset team is in place for those accounts that are very innovative and mature. Here innovative topics like big data can be introduced (C4-I3).

Interviewees C4-I4 pointed out the importance of including the asset early when asked about the last possible time to include the asset. If the PoC had not been successful at the beginning of the client interaction, the project team would not have won the bigger project.

Systematic Reuse

When asked about the maximum reuse, interviewee C4-I3 considered the cross-industry version of the asset to be *reused to the maximum*. It was created in the 1990s in an earlier version and has been refreshed and enhanced with innovations over the last few decades. The interviewee described his service line as robust for decades. “The need for these assets has been there for a long, long time. They will continue to be there for a long, long time” (C4-I3, L20).

Upon verification of what he sees as *maximum reuse*, interviewee C4-I3 explained that almost all projects above a set clip level in this service line are assessed to see if the team can use the assets. The objective is to identify an asset that will differentiate the solution from those of the competitors.

Interviewee C4-I3 (L89-99) described *the team required for systematic reuse*: “The way we're structured is we have an asset buil[d]ing team ... and then you have a solution team that's separate from the asset build team ... The third leg of the equation would be the ... launch advisor”. This combination of asset, solutioning and asset launch team exists alongside the actual project execution team (C4-I3). The interviewee assumed global leaders or the North American leadership decided this setup.

Interviewee C4-I3 is part of the solutioning team. He described his role (L90-92): “I kind of *marry up the assets*, the method to the execution around the programme. So, I'm ... what we call a project launch leader. So, when we sign a deal, I will go in, and I'll basically bring all the assets, tools, methodology and help the programme management launched a programme with[in] the next six to eight weeks ... I'm kind of the bridge between programme execution and presales and ... the use of our method and tools and ... assets and accelerators.”

Reusing this asset is very *systematic*. From a technical perspective, to build the system, the team has well-documented procedures that describe the solution and give a good starting point for applying the asset to any local project environment. From a functional perspective, the business processes are available to leverage the asset (C4-I2).

Asked about *systematic reuse*, interviewee C4-I4 reported that they installed one dedicated project manager responsible for harvesting lessons learned and assets from the programmes that had already deployed the asset. His task is to harvest “with the intention

of baking those learnings into the new releases” (C4-I4, L80). As the technology evolves, the team is upgrading technology and assets, infusing them with the learning of previous programmes.

The asset team performs a *maturity assessment* with their clients (C4-I3). They assess where the client is in terms of capabilities, where the client wants to be, and what the gap is. The outcome of what the asset team calls rapid discovery is a presentation to the client in the form of a business case, including KPIs from a financial perspective. It compares the client to his industry peers (C4-I3).

Figure 4.27 shows the timeline of the reuse processes for Case 4 described in Section 4.5.

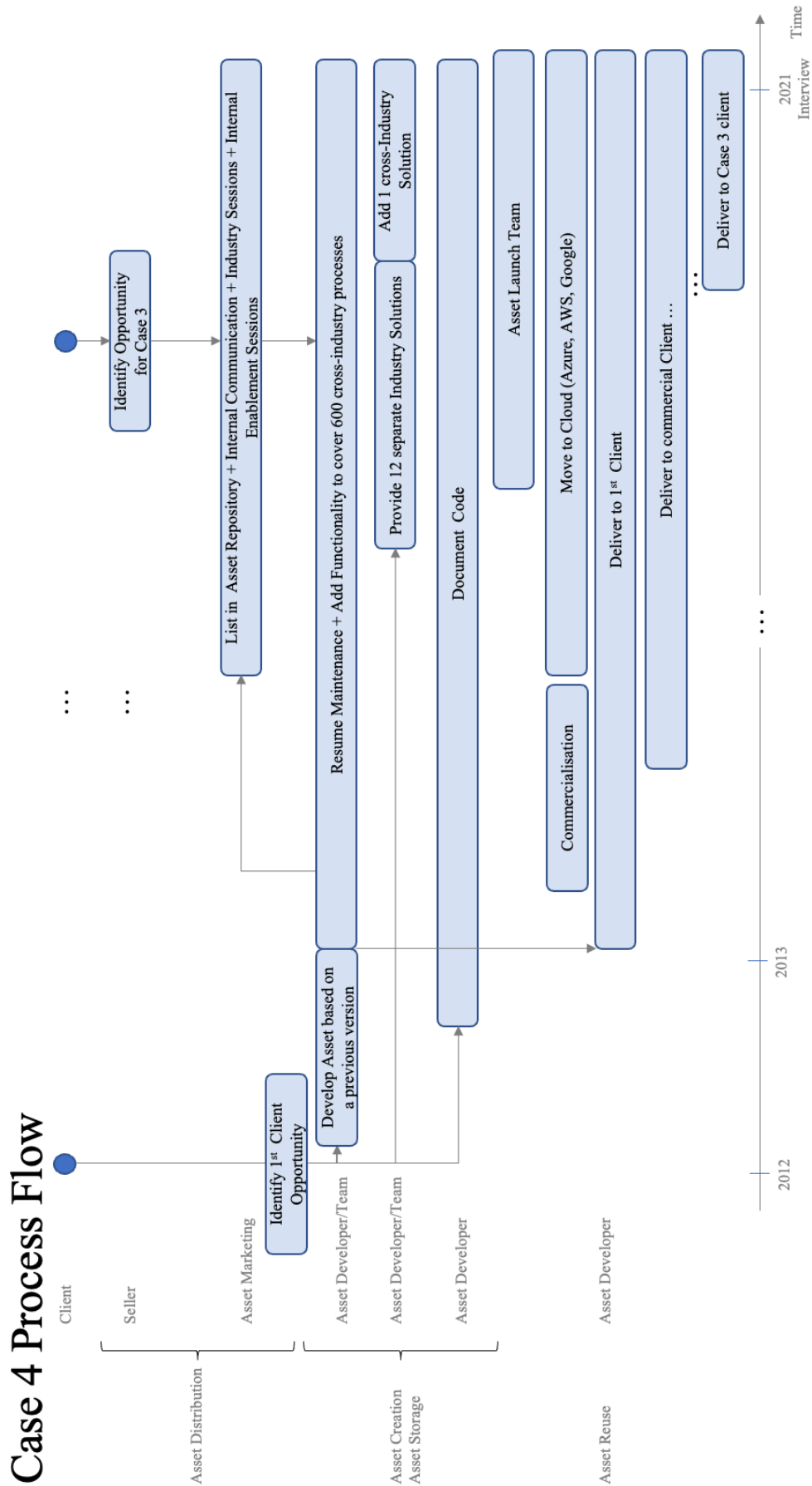


Figure 4.27 Reuse and process timeline for reuse Case 4 (source: author)

4.6 Findings Case 5 – Reuse Initiated during Delivery Generating Savings

In 2016, a small team of experts in India created a software asset for test case automation. Multiple clients have reused it. Case 5 describes how the asset reuse was suggested and implemented by a small team of testers during the delivery phase of an IT project to a Canadian travel & transportation client - see Figure 4.28.

The *client* releases bi-monthly changes on his public website followed by time-consuming regression tests to make sure his website still works. The idea was to automate this testing by using a Selenium testing platform asset (C5-I4).

Reusing the software asset for testing has resulted in more test cases due to shorter testing times, giving testers extra time to generate additional test cases, ultimately *enhancing product quality*.



Figure 4.28 Case 5 - Location of asset owner versus asset reuser - adapted from (Geology.com, 2021)

4.6.1 Case 5 Asset Creation

Figure 4.29 presents the asset creation topics that the interviewees raised in Case 5. These will be discussed now.

Discussion topics	Case 5
Asset creation	
The asset itself	...
Competition	...
Asset team	...
Funding	...
Awards/motivation	...

Figure 4.29 Discussion topics of Section 4.6.1 (source: author)

The Asset Itself

Automation and software products/tools leave a gap in the IT landscape. Here frameworks in the form of software assets can step in. These assets can either be open source or industry-specific and owned by an IT provider (C5-I1). Interviewee C5-I1 decided to create this asset after working with many individuals who had tried to find individual solutions to this framework topic and then got stuck, asking for help. “They go build something on their own without any architectural design or a framework design ... they come back saying ... My framework is not supporting in a right way ... I don't have the integration module. So, when there is a problem, they get back” (C5-I1, L16-20). This experience motivated a handful of testing experts in the test automation centre in India to create the software asset. Interviewee C5-I3 shared the same view and highlighted that many testers do not have programming skills and therefore struggle to learn the scripting required for the task. Therefore, providing a software asset that is mainly scriptless is of great help to a team of reusers – the testers in the field (C5-I3). The reasons for setting up the asset resonate with the reasons featured in Case 3.

Competition

Before creating the asset, the asset owning team *checked if similar assets were available*. There was software available from the various accounts, and there was open-source software as well as software from other providers. However, none of them solved the problem the way the asset owning team had in mind (C5-I1). That is why the asset in Case 5 was created.

Constantly, there is new *software similar to Asset 5 coming up*. Both their quality and their lifetime are unclear (C5-I1, L60). Interviewee C5-I1 (L61-60) explained, “the way

we have built the asset is that it can be reused as the function, ... as the script, as the framework. And within the framework component, ... – it [is] reusable across all the technologies what we update.” So, what the asset owning team created is broader, richer and will be maintained by an expert team once for everyone.

The asset owners deal actively with competitive software assets. Due to the ongoing situation that account teams present their own software asset for the same function and ask for help when they are stuck, interviewee C5-I1 reported introducing *a framework assessment bot* which analyses the software the accounts have built. Depending on the score, this bot recommends whether it makes sense to enhance the self-developed software or skip it and take the software asset instead. This way, the asset owning team has transformed into even better experts in their area of knowledge.

Asset Team

Similar to Cases 1 and 2, after looking for the asset for several years and sharing the responsibility for it, the asset team feels *like a family* (C5-I2).

Asset owners feel very protective about their assets. Asked who invented the asset, interviewee C5-I1 said: “Initially, it’s my ... *babybrain*”. Being a non-native speaker, she uses “babybrain” instead of brainchild” to symbolise her close relationship to the asset. This is in line with previous interviews on assets when interviewee C2-I1 used the term “brainchild”, and interviewee C1-I1 used the term “baby”. Like a baby, the asset needs dedication and input to grow.

There are two major roles within this asset team: experts on the technical side and experts on the deployment side. “The deployment team goes to the accounts and understands their need and comes back to the technical team. And the technical team works on the changes” (C5-I3, L349-350). The deployment team works closely with the offering team (C5-I3).

The three experts working on the asset have *additional roles* and responsibilities in parallel, e.g., custom scripting or development work. Despite this, interviewee C5-I2 characterises the team as very stable.

The asset is sponsored by the global test automation team (C5-I2). The testing centre of competency supports the asset.

Funding

There are no charges for the actual reuse of the asset. The only *charge* is for any labour in developing or adapting the asset (C5-I1, C5-I3). This is how the asset owning team members fund themselves.

Awards/Motivation

Asset owners think holistically – sharing and caring. Like Case 4, Team 5 thinks that one capable, centrally maintained asset is best suited to improve productivity and return on investment (C5-I1). Interviewee C5-I1 has not received any award or similar for creating the asset. However, her manager strongly motivates her intrinsically, allowing her to further develop herself as a person and as an IT expert pursuing an accelerated IT career.

4.6.2 Case 5 Asset Coding and Storage

Figure 4.30 presents the asset coding and storage topics the interviewees raised in Case 5. These will be discussed now.

Discussion topics	Case 5
Asset coding and storage	
Development work	...
Code	...
Knowledge	...
Forgetting knowledge	...
Partnering	...
Innovation	...
Automation	...
Patents	...
Cloud	...

Figure 4.30 Discussion topics of Section 4.6.2 (source: author)

Development Work

Building such an asset takes 12–18 weeks. Over the five years of the asset’s existence, the asset was accompanied by related brother/sister assets for mainframe, mobile and web testing based on client opportunities which arose (C5-I1).

The actual development of the asset was completed by *a technical team* of four to five people. This aligns with the group's explicit knowledge discussed in Section 2.4.3

(Nakamori, 2020; Nonaka & Teece, 2001). The technical expert team conducted brainstorming sessions to develop the minimum viable product plan and the work breakdown structure. Further, there was a coding team involved. That team is the same in charge of deploying the asset (C5-I1). Due to the small team size, team members have multiple roles.

Code

The asset is a testing framework. It does not contain traditional code but consists of a *complete package of Visual Basic VBA scripts and Excel files* (C5-I2).

The asset is fully stored in GitHub. Typically, the asset owning team provides a demo to the team interested in reusing the asset. If they go for reuse, the asset owning team grants the reusing teams of the IT provider access to the asset record in GitHub (C5-I2). *The reusing teams can download the asset* on their own. This is unusual and different to the assets previously considered.

The asset has grown over time. Interviewee C5-I1 shared a long list of asset-specific benefits and application types, this software asset can support, and environments in which this asset can be run. All these new features have been added and announced over the past few years. Up until now, the asset has been updated once per month, sometimes every two months (C5-I2).

These asset updates lead to new versions (C5-I2). The updates are undertaken by three experts in close contact with each other to discuss them. There is no forked development, “only one update” at a time (C5-I2, L148). Any newly created features are typically *shared with the existing set of clients* (C5-I1).

During development, priority is given to generic new features that *bring benefits to multiple rather than just one account*. Accounts are charged for the labour cost required to enhance the asset (C5-I1).

The asset owning team follows a *development plan* when expanding the asset. The asset owning team owns this plan. However, it is closely tied to the needs of the account teams interested in reusing the asset (C5-I1). This software asset is not domain specific, i.e., every industry uses the same asset (C5-I2).

Whenever new features are being added to the framework, the user guide is expanded to document the change. This *code documentation* can be found in GitHub. GitHub contains all the files related to the asset (C5-I2).

The reusing team cannot change the asset's code in the repository. However, when they download the asset to their local folder, they have to make adaptations and can make *changes to their local asset copy* (C5-I2). Usually, these changes do not impact the main framework. The asset owning team advises reusers not to change the main framework. All newly introduced changes typically affect updates in the function files (C5-I2). To do these efficiently, the reusing team receives training from the asset owning team. Sometimes, the reusing team creates functions that the asset owning team later integrates into the main asset and shares with future reusers, e.g., specific government-requested reports (C5-I2). Case 5 is the only reuse case in this case study where experts outside the asset owning team are allowed to adapt the asset. The deal has already been closed, so the asset owning team cannot be paid to make changes. However, reusers must make changes to their local copy of the software asset themselves to enable reuse, which is in line with the dynamic capability theory (Marrucci et al., 2022).

Knowledge

The *knowledge contained in the asset is both technical and business knowledge*, as described in Chapter 2 (Di Cosmo & Zacchiroli, 2017; Dreyer & Wynn, 2016; Huang, 2019; Kneuper, 2002). First, the asset team comes up with an architectural diagram and flow chart. Then they list the technology and the dependencies to consider. Some clients are particular and have tight specifications; they do not accept tools or technologies other than those specified in their company blueprint (C5-I1). The amount of business knowledge required to set up this asset is low. The focus is more on technical knowledge. For example, when the asset team deals with a mainframe application, they then capture mainframe-related processes. There is always a lot to learn as new technologies are constantly being developed (C5-I2).

The asset owning team learned how to create their latest assets by participating in *contests* like hackathons, where they were exposed to new scripts like Cypress or other new technology languages (C5-I1).

Interviewee C5-I1 proudly shared how she came second during a recent *innovation competition*. The structured approach and the templates provided helped her with her asset. Participating and winning this contest motivated the asset owning team and helped spread the news about the asset.

Forgetting Knowledge

“Of our assets, we *keep everything* – it’s there in the Git. ... We don’t forget. It is documented with an installation guide, user guide, anything” (C5-I1, L317-320). This asset team aims to retain all the knowledge and, in contrast to Section 2.4.5 (Spender, 2008), not to forget anything. This way, the asset owning team is constantly increasing their knowledge.

Partnering

The asset team *partners with other IT providers*, e.g., Oracle, on this asset. This turns the asset into a comprehensive but easy-to-apply solution. The asset must follow Oracle cloud release cycle updates (C5-I1) as a side benefit. This has several advantages as it keeps the asset fresh. On the other hand, it creates another dependency that requires regular updating work. The asset team also partners with Open-Source software like Cypress for the asset’s front end (C5-I1).

The asset team collaborates with *clients* on the asset. Client work brings many new aspects that improve the asset (C5-I1). Working closely with business clients over a longer time can be viewed as a partnership.

Innovation

Because the software asset is innovatively applying test automation, it provides a high return on investment (C5-I3). This confirms what was said in Section 2.4.5 – knowledge can be conceptualised for the automation (Succar & Poirier, 2020). “*Innovation* is what we have done ... new on top of what we have ... based on some of the ... requirements from the client.” (C5-I2, L294-297). The interviewee then described how he was able to accommodate two requirements in a smart way into the asset – an approach that had not been taken before. He explained that the team strives very hard to fulfil the client’s requirements by being creative, e.g., using scripting languages like Java. However, there are limits. This is when the team has to decline a requirement.

The asset team *is not using artificial intelligence* (AI) during the coding or storing of the asset (C5-I2).

Automation

The asset team owns an *automation* asset. However, they also use automation for script generation. This is done via Excel VBA. “We record the steps, and we generate the scripts” (C5-I2, L417). They then convert the script into the framework format that makes up their asset. This reduces the team’s time for script creation (C5-I2).

There are situations when the standard way of scripting is not enough, e.g., for implementing test *automation* on mainframes. These cannot be simply automated using Unified Functional Testing (UFT) due to complexities. The team created a special approach by building the UFT on top of some JavaScript and conditional level processes as a workaround. By doing so, they expanded the use of the asset (C5-I2).

Patents

For this asset, no patents have been filed (C5-I2).

Cloud

Plans exist to bring the software asset into the *cloud* (C5-I2, C5-I3). The main difference will be that the framework will be stored on the cloud, not on a local drive (C5-I2). One of the biggest issues is converting the asset into the Selenium framework and vice versa (C5-I2). New scripts must be written to make the asset independent from the current technology so that it can be brought to the cloud (C5-I1). Once the asset is on the *cloud*, it will be easier to deploy onto the virtual machines of remote desktops (C5-I3).

The asset was initially designed for desktop testing only (C5-I1). It would have been better to set it up initially via C# dotnet for mainframe for Windows to make it a cloud asset that could handle Azure and DevOps and allow dockerisation and containers in the cloud. Successful software assets have a long life and need to undergo many transformations.

4.6.3 Case 5 Asset Distribution

Figure 4.31 presents the asset distribution topics that the interviewees raised in Case 5. These will be discussed now.

Discussion topics	Case 5
Asset distribution	
General marketing	...
Internal marketing	...
External marketing	...

Figure 4.31 Discussion topics of Section 4.6.3 (source: author)

General Marketing

The software asset featured in Case 5 serves *multiple use cases*. The automation team created a new database that focuses purely on the served use cases resulting from the successful reuse of automation assets and methods, not on the asset itself (C5-I3). It is a powerful documentation demonstrating the proven reuse capabilities of assets. It could be used to update the IT company’s asset repository with the reuse history of those assets that do not generate signings but generate savings.

Asked about triggers *for including the asset in a client project*, interviewee C5-I4 described situations when a high number of test cases need to be tested repetitively every fortnight and how this is ideal for reusing the identified asset. This is one example of a use case.

Internal Marketing

Asset marketing is done by several people who know the asset well. It typically goes along with a demo and capabilities presentation. Depending on the size of the opportunity, top management can be involved, commenting on the future strategy and new technologies (C5-I1). The asset owning team reports on the asset in internal *newsletters* (C5-I3).

The asset is mapped against a bigger testing *offering* (C5-I1).

The *asset owning team actively approaches accounts*, understands their IT landscape and proposes the solution to increase the asset reach. This is possible because the asset owning team runs regional, sectoral and account-level sessions. They deliver webinars or roadshows where they present their software asset and provide company-internal education on the asset (C5-I1, C5-I3). “We kind of connect with all the accounts ... across sectors where ... we have test presence” (C5-I3, L71-72). If an account has no reasonable

test presence as such but has a particular landscape, the asset will be proposed as well (C5-I3).

Reuse prerequisites can be applied as a search criterion. The asset owning team is informed about the test presence of the accounts by consulting a project database record master list which contains information about the split between technologies, and the current resourcing mix, i.e., the number of testers versus the number of developers. Based on this input, the asset owning team prioritises accounts with more than five testers – slightly different rules apply to SAP-related accounts. The database record master list outlines accounts with a presence in India (C5-I3), which represents the standard situation and covers most of the cases. This means that this approach misses accounts in the Americas or Europe which do not have a testing presence in India. However, *translating reuse prerequisites into opportunity search criteria* is a step towards systematic software asset reuse.

The asset owning team reaches out to those accounts that meet the *pre-set reuse criteria*. If an account is not interested in the asset, but its landscape supports the assets, the asset owning team will still provide the account with a detailed tour of its capabilities and the benefits of asset reuse. “At the same time, we have *sector talks*; we talk ... regularly, so we also tell them that this is the capability that we have, and they can circulate it to their account execs, VPs and all and ... and then come back to us” (C5-I3, L88-89). In addition, the asset owning team runs calls to reach sector and industry experts, account executives and delivery project executives (C5-I3).

The *solutioning experts* should know about the asset if they have the opportunity to propose it. Typically, the solutioning experts will suggest a larger test solution. Asset 5 only covers a subset of the required functionality. The solutioning experts initiate more asset reuse cases than the sellers. During solutioning, more insight exists, allowing the account to work closely with the asset owning team to customise the asset to meet the account’s needs (C5-I3).

Regarding asset reuse, it is estimated that in 60% of the cases, the asset team located the opportunity and pushed the asset; in 40%, the account team located the asset and pulled it in (C5-I3).

There is a sequence in how the asset owning team distributes their asset. They first talk to the client-facing account team, then the solutioning team, and finally conduct a feasibility study for six months (C5-I3). On top of this, the asset owning team delivers *webinars* that target everyone, not just specific communities or teams (C5-I3).

External Marketing

The asset is represented in at least one client centre, the testing client centre in India (C5-I3).

Representatives of the asset owning team participate in competitions to present and share the asset on a wider basis. However, external communication, e.g., on YouTube or Twitter, does not currently occur (C5-I3).

4.6.4 Case 5 Asset Reuse

Figure 4.32 presents the asset reuse topics raised by the interviewees in Case 5. These will be discussed now.

Discussion topics	Case 5
Asset reuse	
Reuse benefits	...
Reuse success	...
Asset reuse	...
Systematic reuse	...
Asset versus product	...

Figure 4.32 Discussion topics of Section 4.6.4 (source: author)

Reuse Benefits

Reusing the asset is beneficial from a cost point of view. The asset has already been reused across *various industries* (C5-I1). Every reuse has provided a proven tool to the client and avoided the creation of brand-new code.

Reuse Success

The number of accounts using the asset can be seen as one *success metric* (C5-I2). However, the asset owning team is not measured against this criterion. Further, because the asset is handed out to the reusing team, the asset owning team is not fully informed about the real number of accounts reusing their asset. If the accounts face an issue with the software asset, they return and ask the asset owning team. Sometimes the asset owning

team performs maintenance on their rolled-out asset. These are situations when the asset owning team witnesses the successful reuse of the asset (C5-I2).

The reuse of this asset is reported to the management in a quarterly meeting (C5-I3). This way, the management is informed about the success of the reuse. *Past reuse success* matters to any new client. It helps to build confidence in taking the deployment forward (C5-I3).

Previous reuses are shared via publishing asset reuse reports. However, interviewee C5-I3 was unaware that past reuse successes are not documented in the IT company's asset repository. This is because the central team who updates the repository focuses on revenue only, but the asset in Case 5 does not generate any revenue. Instead, it leads to comparatively small *savings*.

Asset Reuse

The asset owning team is onboarding, on average, one new asset reuse client per quarter (C5-I1). However, real reuse is unpredictable and fluctuating; it is not constant and cannot be planned for (C5-I2).

Interviewee C5-I4, a member of the IT provider's account team and in charge of testing, had heard via e-mail campaigns run by the Test Automation community that certain offerings comprise software assets that can help automate regression testing. When interviewee C5-I4 turned to the test automation team owning the testing asset, he did not have a specific asset in mind. Instead, he gave them two or three use cases and asked if they thought these could be automated. Only one asset was *applicable* to the given situation, not multiple assets to choose from. The asset owners helped to create a prototype for demonstration and assured him that the time spent on regression testing could be reduced by automation.

Creating a prototype and implementing some use cases was a way of assessing the asset. Circa four experts from the asset owning team and four experts from the IT provider's account team were involved in preparing this particular asset reuse featured in Case 5 (C5-I4). Clients are cautious about new software. The success of reuse needs to be demonstrated upfront.

Before the reuse, the required infrastructure had to be set up on the client's network to maintain the high-security level. Once the reuse began, Asset 5 was used externally, by the client's testing team, and internally, by the testing team that is part of the IT provider's account team (C5-I4).

The client decided to reuse the identified asset based on the expected benefit. The effort had to be calculated and presented to the client for this. The effort comprised elements like test automation training, implementation effort in terms of environment setup, and contractual changes resulting from this reuse, including defining a specific work order (C5-I2, C5-I4).

The client decided to go for a software asset owned by the IT provider but is *open source*. According to interviewee C5-I4, his client would have also accepted a pure open-source asset. However, other clients could be cautious about open-source software depending on the domain due to security worries.

Up until now, the asset has been used in its *initial configuration*. However, the asset owning team has offered updates resulting from their work with peer clients – should these be required (C5-I4). In Case 5, there is a less tight collaboration between the asset owning team and the asset reuser because the account team adapted the asset for reuse.

In contrast to other assets, this test automation *asset does not involve a deadline for* when it is too late to initiate reuse. As long as there is further testing to be done, e.g., regression testing, it makes sense to reuse this testing asset (C5-I3).

Reusing the asset is not difficult but requires some previous testing experience and hands-on *training* on the asset. Just referring to documentation would not be sufficient. Attending a virtual classroom and properly creating test cases is important. The asset was installed by the asset owning team on the reusing team's machines (C5-I2). This confirms that often explicit, written knowledge is insufficient for reusing the asset (see Section 2.4.3 for details) (Nakamori, 2020; Nonaka & Teece, 2001). The virtual classroom gives the reusers a chance for interaction to tap tacit knowledge.

The fact that the *asset owning team has passed the asset beyond their control* and let the client and the IT provider's account team operate and manipulate it, is atypical. The reason behind this is, the fixed-price contract with the client had already been closed and therefore it would have been very challenging to onboard additional testers. The client

would have expected that for free. In addition, the client made it clear that he did not want to carry any maintenance cost. No matter what suitable offering or asset emerges, after contract closure, the client will not accept any changes to the fixed-price contract (C5-I4).

Interviewee C5-I4 sees *the point in time* he suggested the asset as perfect – not too early and not too late. It was not too early because he had the chance to see how the client’s team perform without the asset; both he and the client noticed the pain and then took action almost immediately. There is a natural end point when suggesting asset reuse: when the contract ends. But even then, reuse could still be beneficial, in particular when discussing a contract renewal.

Finally, *three people from the IT provider’s account team are reusing the asset for the client*. So far, the asset has performed as promised. The time for each test run has been reduced from 2 hours to 20 minutes. This can be seen as a success metric. At the time of the interview, about 50 test cases had been created using the software asset. If a new function is added to the client’s website and has to be tested, this always requires a new automated test case to be set up, using the test automation software asset (C5-I4).

Reusing the asset has not yet led to any noticeable *innovation*. So far, the asset is only used for one application (C5-I4).

Overall, interviewee C5-I4 perceives the reuse of the asset as a good experience which he would recommend to others. Reusing this asset *has increased the knowledge of the asset reusing team*. This aligns with the literature (Barros-Justo et al., 2018) as discussed in Section 2.5.5. They have learned more about related software assets for testing and about alternative offerings. Reflecting on this, the current client opportunity was not properly mapped against an offering (C5-I4). Otherwise, the suitable asset could have been identified earlier. In contrast to what Maccanti (2016) writes (see Section 2.5.5), it is clear where the asset has been reused, how it has been reused and why.

Systematic Reuse

For interviewee C5-I1, *systematic reuse* means that the architecture, most of the components the team built, the overall architectural asset design, the master script and the data sheet adopted are still being reused.

Systematic reuse happens “the more accounts we connect” (C5-I2, L255). A feasible opportunity exists if accounts use UFT and seek a new framework. The asset deployment team, consisting of a handful of people, performs an account assessment to identify those needing the asset (C5-I2).

Asset versus Product

Asked about why the solution is a reusable software asset and not a product, interviewee C5-I2 explained, “it doesn't actually work on its own. It is on top of the UFT tool, which is a UFT product ... There are ... some set of functions ... works along with UFT. So, I think that is why it is called an asset” (C5-I2, L321-327).

Figure 4.33 shows the timeline of the reuse processes for Case 5 described in Section 4.6.

Case 5 Process Flow

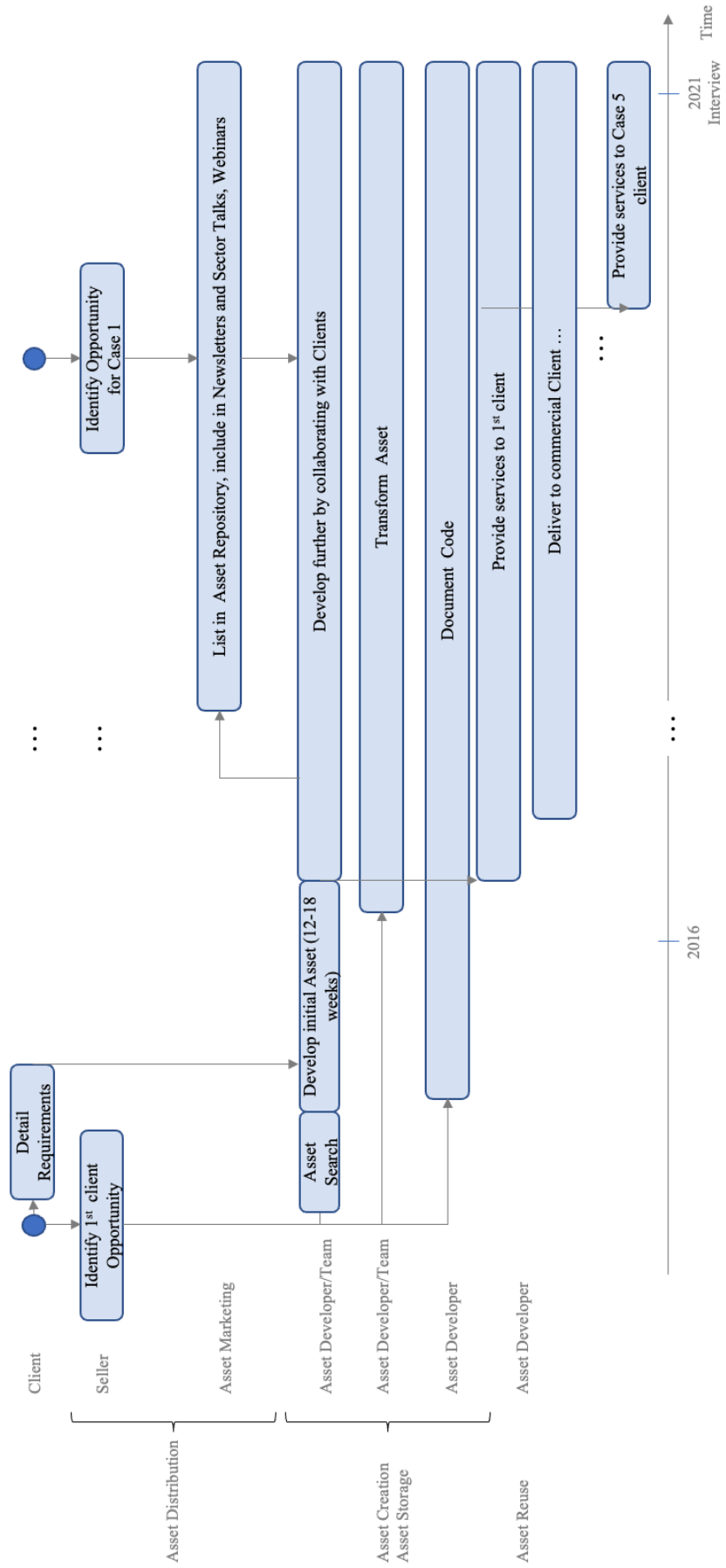


Figure 4.33 Reuse and process timeline for reuse Case 5 (source: author)

4.7 Case Study Discussion

Contrasting and comparing cases via matching patterns or providing rival explanations delivers a condensed and more explicit case study result. This section discusses all five previously covered cases and puts them in perspective.

For more structure, all cases are discussed following the identified software asset reuse phases – Asset Creation, Section 4.7.1; Asset Coding/Storage, Section 4.7.2; Asset Distribution, Section 4.7.3; and Asset Reuse, Section 4.7.4 respectively.

4.7.1 Asset Creation

All cases 1-5 focus on two main asset creation topics – the asset itself and the asset team (Figure 4.34). These topics will now be discussed and compared across all five cases.

Discussion topics	Case 1	Case 2	Case 3	Case 4	Case 5
Asset creation					
The asset itself					
The asset team					

Figure 4.34 Discussion topics of Section 4.7.1 (source: author)

The Asset Itself

All cases demonstrate a clear, often case-specific, objective for setting up a reusable software asset. What all assets have in common, not mentioned in literature yet, is that they *close a business gap with one high-quality software* that is gradually built and *accumulates the knowledge of experts* over the years. This enriches the resource-based view refined by Barney (2001) by an important time aspect. Knowledge is not solely generated by a singular, valuable resource or a team of resources within a restricted timeframe; rather, it accrues over time. Teams inherit specialised asset knowledge from preceding teams and expand it progressively over decades.

Checking for already existing software should be a necessary step before creating new reusables. In contrast to Stenholm et al. (2019), all teams consciously checked what was available before setting up a new reusable software asset, proving their status as real reuse professionals. This finding contributes to the extant body of knowledge on the processes involved in reusable software development and holds significant implications for practical applications within the field.

The asset owning teams build up knowledge. This confirms the power of small dedicated expert teams as described by Zhou (2019), Human (2020) and others in Section 2.4.3 as well as Becker's human capital theory underlining employee development (England & Folbre, 2023) when it comes to the creation and setup of reusable software assets. The literature describes how learning turns teams into a centre of competence (Irick, 2007; Ram & Devi, 2019) but misses that asset teams, because of the asset, become a human capital asset by forming a centre of competence that is of value in the wider business. The finding that a reusable software asset has a bundling effect gluing experts with related skill sets virtually together, even if they are distributed over the organisation, enriches the theory of knowledge creation in a business relevant way.

All five cases confirm that software contains knowledge. Starting to use this reusable software asset, new clients benefit from their own knowledge and the IT provider's developer knowledge. For the first time, this case study records that clients benefit from the knowledge of all those clients who earlier signed up for reuse and contributed to this asset. This *additional peer-client contribution to the knowledge contained in the asset* was a selling point for the client in Case 2. This finding notably enriches the theory of knowledge work by incorporating a new knowledge-contributing group – the set of current clients. Within the context of knowledge work, as elaborated in Section 2.4.3, scholarly attention has predominantly been directed towards individual knowledge creators and the groups they constitute, see SECI-concept in Figure 2.12. This case study suggests that, in the realm of software asset reuse, there exists more than just one group of knowledge creators.

Asset owners must keep up with the *competition*. After all asset owning teams reported external asset competition, it could be deductively assumed this would apply to internal competition as well. Indeed, Assets 1, 3 and 5 owners reported ongoing internal asset competition. This was also the case for the predecessor of Asset 2. The new Asset 2 is large, internally used and therefore well known to all employees of the IT provider. So far, it has not faced any internal competition, but this could change over time. Asset 4, which is big and well-known in the respective internal teams, has, up until now, seen only one competitive software asset that was much smaller in scope. The interviewees identified two strategies for managing all forms of competition: partnering with owners of the company's competing software assets and exploiting the synergies with related assets, as described in Section 2.2.2 by Haskel & Westlake (2018); this gives them a more

substantial market. No discernible scholarly sources were identified that specifically address the internal competition of software assets and the subsequent supplementary costs. As a result, this finding serves to augment the research conducted by Haskel & Westlake (2018) by incorporating the consideration of intra-organisational competition within the area of software asset reuse.

From the perspective of the knowledge content discussed in the literature section 2.4.2, the case assets absorb both the asset-developing team's and the client's knowledge. Therefore they fulfil a knowledge-related role confirming Haskel and Westlake's theory (2018). They are based on software and related artefacts.

The Asset Team

Reusable *software assets* are typically created, maintained and *owned* (in the sense of responsibility) *by a team*. Previous research provides the impression of reusable software assets sitting unmanned on a shelf, waiting to be reused. Instead, the featured software Assets 1–5 are owned and actively managed by dedicated teams of different sizes and roles. All cases benefit from strong asset sponsorship. The management of relevant internal organisations sponsors assets 1, 4 and 5. Assets 2 and 3 have dedicated asset sponsors who are senior executives passionately supporting the respective asset. All asset owning teams show some unexpected characteristics, which are discussed next. As no literature could be found on asset owning teams, this research enriches the knowledge on software asset reuse in general and the research of AL-Badareen (2021), in particular, who discusses transition operations without commenting on the team doing this work. The following sections detail the knowledge gained about the teams who create and maintain reusable software assets.

Asset owning teams have a very tight internal relationship. Asset Team 2 labelled their team as a start-up, given the freedom they have to try out new things.

Asset owning teams are unusually *stable*. This is unexpected because IT people change their jobs statistically more often than others (Statistics, 2020), as discussed in Section 4.2.1. Applying deduction, it could be assumed all asset owning teams behave the same. However, it must be differentiated. Smaller teams tend to keep their team members until retirement, while bigger teams see fluctuation, but attrition rates are much lower than

regular. The working conditions within an asset owning team seem to be more attractive compared to other setups.

Asset owning teams are strongly motivated. This could be one reason why their attrition rate is minimal. Case 1 interviewees reported how the business success of their asset provides autonomous job *motivation*. Case 2 interviewees felt motivated by contributing end to end in helping the client in addition to having the chance to try out new methods and technology. Other teams were pleased with the asset's high visibility within the IT provider's organisation. Helping the client was identified consistently by all teams as a motivator.

When asked about *awards*, all interviewees needed some time to remember. This aligns with Kuo's (2013) findings mentioned in Section 2.4.7. Teams are not reward-driven but have other sources of motivation. Awards do not significantly contribute to the teams' motivation in software asset reuse. Mapped against Vroom's expectancy theory (Lloyd & Mertens, 2018), asset owning teams are intrinsically motivated.

Learning is a consistent pattern within all asset owning teams. All teams indicated how the effect of learning directly impacts their reusable software asset underlining Teece' dynamic capabilities (Marrucci et al., 2022) theory. Permanent learning has been described by Burciu & Kicsi (2015). Further, the asset owning team itself forms a valuable asset by acting as a centre of competence as described in the literature (Irick, 2007; Ram & Devi, 2019).

Team funding is a direct measure of asset reuse success. Assets 1, 2 and 3 are funded based on external and internal reuse of their asset. Assets 4 and 5 are funded through the claim codes they receive. Asset teams can be seen as financially independent; success allows them to grow their team. Whether the knowledge exploitation (Section 2.2.2, Figure 2.4) is in the form of income or efficiency savings depends on whether the asset reuse occurs in an intra-organisational or inter-organisational context (Section 2.4.4). While intra-organisational reuse settings focus on internal efficiencies so that teams are funded through claim codes, inter-organisational, external reuse results in company revenue.

Because of the lack of real reuse cases in the literature, other previous research overlooked the importance of the asset owning team's fundamental role. The findings of this thesis

acknowledge the crucial aspects of self-motivation, self-funding, and continuous learning within the asset owning team, which are essential for the success of software asset reuse. This expands the theory of knowledge in the area of knowledge processes, in particular in the realm of creating reusable software assets.

Figure 4.35 summarises the case study findings related to asset creation, which have been discussed above and can be generalised across all five cases.

Asset Creation

- **Stable**, self-funded, highly-motivated, constantly learning, family-like internal “start-up”-**teams + clients** create **assets containing MORE knowledge** than one-off software.
- Partnering helps dealing with competition
- Awards don’t motivate!




Figure 4.35 Summary of case study outcome – asset creation (source: author)

4.7.2 Asset Coding and Storage

All five cases 1-5 are centred around three main asset coding and storage topics: development work, continuous changes and innovation (Figure 4.36). These topics will now be discussed and compared across all five cases.

Discussion topics	Case 1	Case 2	Case 3	Case 4	Case 5
Asset coding and storage					
Development work					
Continuous changes					
Innovation					

Figure 4.36 Discussion topics of Section 4.7.2 (source: author)

Development Work

The *asset owning team creates, changes, and maintains* the well-documented reusable software asset. In line with Gamble (2020) and Majchrzak (2004), the asset is available as explicit knowledge. Adding to the body of knowledge, the development and further

maintenance of reusable software assets is typically in the hands of the asset owning team. This indicates a great proportion of tacit knowledge which cannot be easily documented (Nickols, 2000) as discussed in Section 2.4.3. Case 5 forms an exception. Here, the asset owning team allows the reuser to make some final changes on a local copy of the reusable software asset because the account team cannot pay for any changes after the contract has been closed. It helps that Asset 5 is, in technical terms, the simplest of all five presented assets. Alternative Case 5-assets are expected to show the same behaviour.

Asset owning teams partner with others. These *partnerships* go beyond the set of clients and include third-party providers, open source or other software assets, confirming Haskel & Westlake's (2018) findings on asset synergies. Partnering strengthens the reusable software asset and often increases its reach. It demonstrates an organisation's ability to learn and change, as described by Teece's dynamic capabilities theory (Marrucci et al., 2022).

As the literature section 2.4.1 identifies, the two *main knowledge media* are humans maintaining tacit knowledge and IT systems storing explicit knowledge.

Software assets are stored in repositories that act as a buffer and preserve knowledge over decades, as discussed in Section 2.4.7 (Abdelwhab Ali et al., 2019; Sahibzada et al., 2019). In addition to the *code repository* covered in most of the literature, additional repositories could exist, e.g., an *asset meta-data repository* or a *use case repository*. This adds to the theory of knowledge processes contributed by Taraszewski (2017) and others. Asset meta-data repositories can be useful for central asset management. A use case repository can reflect all reuses of the asset in terms of specific use cases and clients. Consistent with the dynamic capability theory (Marrucci et al., 2022), the evolving nature of the asset necessitates effective management to ensure the currency of all asset repositories. Company-wide one-place-to-go repositories can achieve the best effect.

The interviewed asset owning teams dismiss forgetting (de Holan & Phillips, 2004; Spender, 2008) – they do not lose any code. Instead, *all code versions are kept* (Di Cosmo & Zacchiroli, 2017). This finding improves the current theory on knowledge storage, particularly the work of Spender (2008). Knowledge is seen as a building. This has multiple effects. Early code from almost 30 years ago is still actively used today, representing a significant wealth of knowledge. However, this also means a substantial amount of code to manage. It is important to examine the underlying reasons, advantages,

and disadvantages. One reason could be that code is kept respecting its earlier and still present developer.

Continuous Changes

Every client 1–5 received an adapted, customised code set of the software asset. In contrast to what has been noted by other researchers, *different asset feature functions are required* because of the typically different experimental situations the software asset is exposed. This confirms the nature of reuse as stated by Cacioppo (2015) and Schloss (2018), described in Section 2.3.1. It can be concluded from this, in contrast to what was said in Sections 2.5.2 and 2.5.5, there is *no reusable COTS for B2B niche* clients. This improves the theory of knowledge processes in the area of the creation of reusable software assets, e.g., the work of AL-Badareen (2021). In future, due to cloudification, developers could go for customisation, which allows them to switch features for specific clients on or off.

The software code is *being continuously enhanced and maintained*. Changes are driven by close interaction with the client, continuous improvement, changes in technology or business, regulations or in response to internal or external competitors. Notably, features created for one client are often offered to the entire set of existing clients. Through this partnership model, clients benefit directly from the ideas of their peer clients. Asset 5 forms an exception – newly introduced features are shared with new clients but not always with the existing clients because the asset owning team does not always undertake software asset adaptation.

The cost of reusing a reusable software asset is not as low as the existing literature indicates (Barros-Justo et al., 2018; Krüger & Berger, 2020; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Sandhu & Batth, 2021b; Van Buren et al., 2016). This enhancement in the theory of knowledge work in the area of knowledge storage acknowledges not only the reduction in costs that reuse offers, but also the supplementary, cost-impacting endeavours that reuse necessitates.

Transforming the asset by *repurposing* it provides access to additional client sets. Horizontal reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010) requires *abstraction*. It adds new client sets to software assets with so far only a vertical reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010).

Cases 1, 4 and 5 reported *sibling software assets*. Unknown in the literature, these are assets where the changes to the original asset would have been too significant to incorporate into the existing code. Hence it was decided to create a separate, closely related software asset, e.g., for a specific industry or topic related to the original software asset. This finding expands the theory of knowledge work in the area of knowledge storage. So, for example, Haskel & Westlake (2018) describe how assets can benefit from one another but have not yet mentioned sibling assets, yet.

Innovation

The above-listed changes can lead to *innovation*. They remove the client's pain and lift the quality of the asset to a new level. This finding confirms the theoretical framework presented in Section 2.7 (Alrowwad & Abualoush, 2020). Introducing automation, AI, agile working, cloud, microservices or other technologies can drive innovation. Innovation happens gradually when the asset owning team responds to a client's or partner's requests concerning the software asset.

The *agile* work model speeds up work. Cases 2–5 are fully operated in an agile way. This helps to achieve fast innovation for the client.

Automation helps reuse innovation and turns the IT provider from a software provider into a service provider. This way, automation drives servitisation.

In addition, cloudification drives the servitisation of reusable software assets. This has simplified the handling of Assets 2–4. Case 1 allows additional analysing services to be provided to the clients. Providing safe cloud services keeps the asset owning teams busy and challenges them to explore and expand the full benefits the cloud can bring.

The IP of reusable software assets is *not patented*. This is consistent across all cases; only Case 2 reports a few patents. This is a surprise because the IP provider is known for its strength in patent leadership and knowledge management (Wijnhoven, 2008). One reason could be that patents have no practical impact on the business success of the individual reusable software asset.

Figure 4.37 summarises the case study findings related to asset coding and storage, which have been discussed above and can be generalised across all five cases.

Asset Coding/Storage

- The **Asset Owners** code and **continuously** maintain, **abstract**, enhance, **repurpose** and **adapt** the software asset in an innovative way (AI, agile, microservices, ...).
- Patents don't protect!




Figure 4.37 Summary of case study outcome – asset coding and storage (source: author)

4.7.3 Asset Distribution

All five cases 1-5 focus on three main asset distribution topics – general, internal and external marketing (Figure 4.38). These topics will now be discussed and compared across all five cases. This section corroborates the research conducted by Wallace (2018). It adds to the theoretical understanding of knowledge work by elaborating on Wallace’ (2018) push actions. This is achieved by providing illustrative instances of the proactive knowledge distribution the asset owners have to initiate in order to enable the reusing team to identify a suitable software asset for planned reuse.

Discussion	Case 1	Case 2	Case 3	Case 4	Case 5
Asset distribution					
General marketing					
Internal marketing					
External marketing					

Figure 4.38 Discussion points of Section 4.7.3 (source: author)

General Marketing

Use cases and *client references* are vital for marketing reusable software assets. The interviewed asset owners emphasised their client references – some have only one strong reference client, e.g., Asset 2. While all Assets 1–5 are listed in an asset repository containing asset meta-data, only Asset 5 is listed in an additional repository presenting each single (re)use case. Sharing identified use cases enables teams to develop new use cases more efficiently. New market niches for reusable software assets can be found by constantly expanding the use cases for reusable software assets.

The asset owning teams of Assets 1 and 4 estimate that their team identifies the reuse opportunity in about 50% of their reuse cases. In the other 50% of cases, the IT provider's account team, representing the reuser, identifies it and approaches them. This is in line with this case study. In Cases 1 and 5, the account team representing the reuser approached the asset owner and suggested reusing the software asset.

Contrary to popular belief, software asset reuse does not happen by itself. It requires *marketing*. There is no central marketing budget available for reusable software assets. Marketing is, therefore, unplanned and unfunded until the asset owning team voluntarily intervenes. Marketing actions target an internal or an external audience.

Both vertical, inner-domain, and horizontal, cross-domain, software asset reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010) require substantial asset promotion. Most reusers do not actively search for reusable software. Even if they would favour reuse over make-new, they cannot assume a software is available that can be adapted to their use case in the niche area they are working on. That is why it makes sense for the asset owning team to push reusable software assets (Wallace, 2018). Nonetheless, the account team's pull of assets necessitates an initial effort from the software asset owner to inform potential reusers about the asset's presence. Reuse thrives on *internal and external marketing* actions. None of these are mentioned in the literature that is focussed on IT only.

Beside humans and IT tools like GitHub, further knowledge media in the sense of Section 2.4.6 are asset newsletters sent to external clients or company-internal discussion chats on Slack.

Internal Marketing

Global companies are organised into account teams. The account team first creates the reusable software asset for their client – who will be the first client to use the asset. To reuse the asset, a new client must be identified. This is often only possible via the respective account team. For Case 2, for example, clients in another domain had to be addressed in a *two-step sales process*. First, the IT provider's account team had to be convinced through internal marketing by the asset owning team. Only then could the asset owning team contact the client team directly. This two-step sales process expands

Wallace's (2018) push approach discussed in Section 2.4.7. Its implications have not yet been mentioned in previous software asset reuse literature.

Internal awareness sessions allow the asset owning team to showcase their reusable software asset to a broader community. All five asset owning teams reported presenting their asset at these sessions. This confirms Wallace's (2018) push model in which the asset owning team must bring the asset to market. Interviewee C3-I4 attended one of these education sessions and first heard about the asset his client is now reusing. Without internal asset awareness sessions, the account teams might not know the asset's existence or its latest features.

The moment a client opportunity arises varies from the moment when the other account teams are trained on the asset. The account teams must remember the asset and are expected to look for a chance to propose the reusable software asset to their client. To close this gap, the IT provider is doing something the literature has not described: *mapping its reusable software assets to its offerings*. Each time an account works on a new opportunity, they must specify the respective offering and are reminded of the appropriate reusable asset.

The assets are documented on wikis, the intranet or various internal platforms. All five presented assets are listed in an *asset-meta data repository* which provides information about the assets. This includes, e.g., links to the wiki sites, the mapping against offerings or the names of the active asset owners. Despite massive efforts, the asset records quickly become outdated due to the fast-changing asset features. Further, potential reusers need help finding and understanding content. In this context, the asset meta-data repository provides the biggest value to the management or the global asset team as it helps people using the repository daily to administer the asset portfolio.

The asset owning team or affiliated organisations send mailers in larger newsletters to direct potential reusers to repository records. These mailers are supposed to act like a *trigger* (see Figure 2.21) and remind potential reusers of the asset. This way, interviewee C5-I4 heard about the asset for the first time. This finding confirms Fogg's Behaviour Model (2009) discussed in Section 2.5.2.

In the past, the solutioning teams sometimes initiated the reuse of Assets 3–5. Therefore, the asset owning teams *work regularly and closely with the central solutioning teams*. If

education on these assets and the methodology they use is included in the education of internal experts, the reuse of the software asset can be increased. This confirms Becker's human capital theory (England & Folbre, 2023).

The asset owning team of Asset 1 runs an *asset-specific slack channel* company internally. This binds the team of owners and reusers together and stipulates vertical reuse, as discussed in Section 2.5.4 (Anasuodei & Ojekudo, 2021; Jalender et al., 2010). This allows knowledge and use cases on the reusable asset to be shared easily.

Competitions are an excellent way to share news on a reusable software asset. Internal hackathons have been used by interviewee C5-I1 to demonstrate the power of her asset.

Opportunity pipelines can be scanned to find suitable reuse opportunities for existing reusable software assets. Additionally, assets typically proposed during the sales stage can benefit from checking the opportunity pipeline because it helps to identify the right set of people that should be internally contacted and informed about the existence of the asset when it comes to future deals. To engage in systematic reuse analysis by scanning records, the asset owning team needs to have an abstract understanding of the software asset's technical and non-technical prerequisites and translate these into search operations that can be applied company-internally.

Software reuse literature needs to mention distribution methods like awareness sessions, asset mapping to offerings or opportunities, and various kinds of asset repositories.

This thesis enriches the theory of knowledge by detailing the teams involved in software asset reuse and their reuse related roles within a global IT company, thus adding to the complexity of the SECI-model in Figure 2.12.

External Marketing

Retaining clients is a success indicator. It demonstrates the asset's strength and the quality of the client relationship. Further, it diminishes the urgency of looking for new clients. At least Assets 1, 2, and 4 still deliver services to their first clients. Asset 1 provides services for 30 years, demonstrating a long lifetime for this reusable software, confirming that reuse extends the lifetime of software (Lear, 2021).

The owners of Asset 1 send *regular newsletters to their external clients*. This ties clients closer to them and allows them to update existing clients on new asset features or other relevant asset news.

The asset owning teams of Assets 1–4 reported working on client *RfPs* whereby they suggested the reuse of their respective reusable software asset.

External marketing activities, e.g., as a member of external communities or associations, help find new clients. External reports can propagate the use of the asset.

Software demos convince clients. Because the software exists, the asset owners can *demo* their asset's capability to a potential client.

The *price* of using a reusable software asset can be quickly provided to the client. The asset owning team has pricing experience.

However, the price for the service is not as low as the existing literature indicates (Barros-Justo et al., 2018; Krüger & Berger, 2020; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Sandhu & Batth, 2021b; Van Buren et al., 2016). The reuse savings are counter-weighted by the high cost of maintaining the asset until the next business opportunity is won.

Pricing asset services is a challenge (Cases 2 and 3). The absence of future clients may increase the pressure to charge the current client too much. The price for the asset service fee should be provided more quickly than the price for new coding (Case 3).

Support services can be offered. *Dedicated sellers* can help to sell software services.

A *reuser community call* allows knowledge exchange and joint brainstorming amongst clients. This ties clients together as a community. It allows the IT provider to share developed software features in a more interactive way at an earlier point in time.

Software reuse literature should list the above detailed external marketing activities.

Distribution connects the asset owning team with the asset reusing team across space and time. Different methods are used to transfer the reusable software *geographically* (see Figure 4.39). Other than what the researcher expected, no single asset distribution method could be identified. Instead, there is a whole tool kit for asset distribution.

This research improves the theory of Stahel (2016) research, reuse can start earlier in the cycle to include intervention via code adaptation and distribution in the form of marketing or sharing meta-information about the reusable software asset (see the blue updates in Figure 4.39).

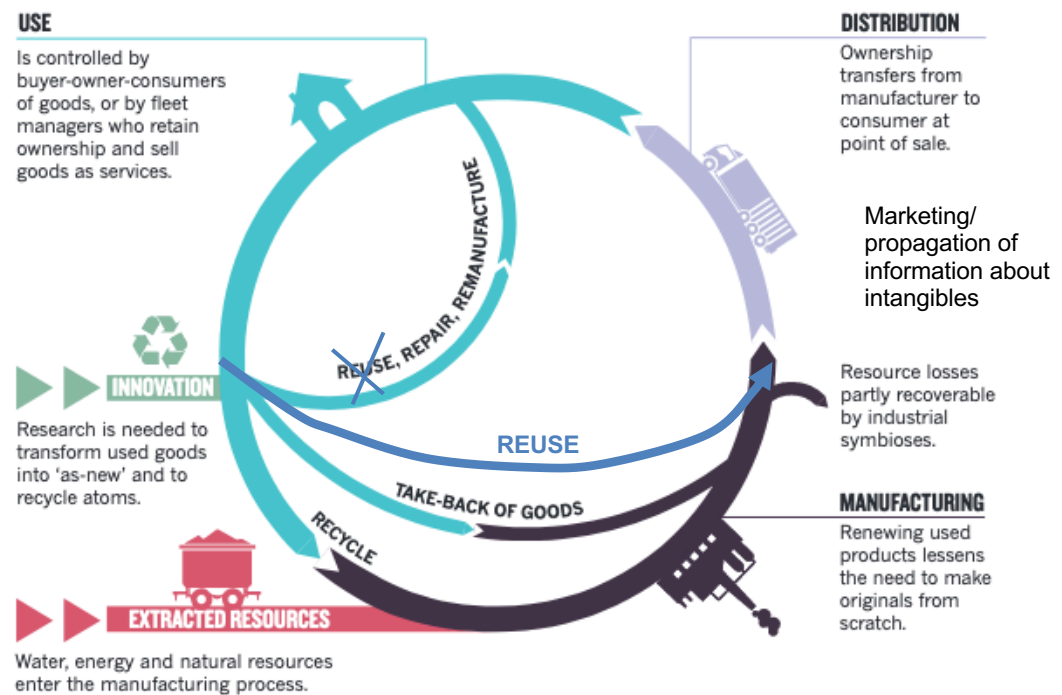


Figure 4.39 Reuse as part of the circular economy - adapted from (Stahel, 2016)

Systematic Reuse

Distribution bridges *times*. Asset reuse can occur asynchronously, e.g., years after the asset was created. Continuously distributing the asset reminds potential reusers of the existence of the asset. Adding to the research of Cacioppo (2015) and Schloss (2018), the present study enhances the current theory of knowledge processes by elucidating the heretofore unexplored impact of the factor time on reuse. Contrary to the rapid pace of many contemporary business practices, reuse demonstrates its efficacy over more protracted periods – spanning months, years, or even decades.

Figure 4.40 summarises the asset distribution-related case study findings that can be generalised across all five cases.

Asset Distribution

- The Asset Owners ...
 - supported by sellers or solutioners
 - use references and abstract use cases to
 - **constantly**, creatively, and systematically
 - **push**
 - 2-step-internally or directly externally the asset to the client

to early map and long-lasting marry their asset to an opportunity.

- Some reusers **later** remember some assets and contact the asset owners asking for reuse
- Asset distribution is driven by humans – not by repositories!




Figure 4.40 Summary of case study outcome – asset distribution (source: author)

4.7.4 Asset Reuse

All five cases 1-5 focus on three main asset reuse topics – asset benefits/reporting/drawbacks, asset reuse and systematic reuse (Figure 4.41). These topics will now be discussed and compared across all five cases.

Discussion topics	Case 1	Case 2	Case 3	Case 4	Case 5
Asset reuse					
Asset benefits, reporting and drawbacks					
Asset reuse					
Systematic reuse					

Figure 4.41 Discussion topics of Section 4.7.4 (source: author)

Asset Benefits, Reporting and Drawbacks

Reusable software assets offer many *benefits* and drive superior business performance confirming the resource-based view refined by Barney (2001). Countable revenue can only be agreed upon during the sales or early solutioning phase. In addition, the asset owning teams reported uncountable advantages like lower development risks and, therefore, lower risk ratings in contracts. The asset team can demo their software to the client ad hoc because it already exists. This is a differentiator in the market and gives the IT provider a competitive advantage (C3-I4). Clients can benefit from innovative asset features introduced to the asset as a result of the request of other clients.

Further, *the reusable software asset serves a niche market* which would otherwise not be affordable. This enriches Zhou's (2001) and Spoelstra's (2011) theory of a reusable software asset. One reason for niche-serving ability is the skill reuse on the IT provider side: the asset team bundles skills that would otherwise be distributed. Improvement ideas from the client can help the IT provider internally. Concentrating on one software asset instead of several small one-off solutions helps the team stay abreast of technology. Another benefit is collaborating with the external user community of the asset towards further improvement.

An open-source approach offers additional benefits. Assets 3, 4 and 5 are free of any licence fee, entirely or partly because they are open source. In this case, the client *only needs to pay specific service fees*.

Reusing software has a few drawbacks. Case 2 reported that the cost is considerably high, even for small changes. They also reported that the client only pays 25% of the development cost. Further, changes need to be prioritised, to the disadvantage of internal clients. Case 3 reported that it always takes time for the asset benefits to materialise, and persistence is required in the meantime. This enhances the knowledge on software asset reuse by adding drawbacks to the long list of benefits (Barros-Justo et al., 2018; Horne, 1995; Majchrzak et al., 2004; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Salgot et al., 2017; Sandhu & Batth, 2021b; Wilson et al., 2000).

All asset owning teams reported *not having received any reuse targets* from management. This is unusual in a target-driven, money-focused IT industry. It contrasts Vroom's expectancy (Lloyd & Mertens, 2018) but can be explained by the intrinsic motivations of asset owning teams, as mentioned in Section 4.7.1. Reuse is less suited to maximising profit – it aims to maintain value within an economic system. This explains why, following a capitalist focus on a linear economy, reuse will not be pushed to the limit. A linear economy and a circular economy embody distinct value systems.

Teams report their reuse successes, e.g., the number of clients per asset. In a target-driven company, the reuse team struggles because no clear success metrics and reuse targets exist.

Asset Reuse

Case 4 has a Single Point of Contact (SPoC) in place who accompanies the reuser team during the reuse. Concluding from all cases, it is worth pursuing all reuse options – both horizontal and vertical reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010).

The interviewed teams see an increased reuse trend over time.

Contrasting Stahel (2016), account teams should focus on *reuse as early as possible* to maximise business benefits. This can be achieved by working closely with the asset owning team and the solutioning team (Case 3). A maturity assessment can be performed with the client to marry the asset to an upcoming opportunity.

Big applications should be fed into reuse in their entirety and as early as possible in the sales cycle – confirming one of KPMG’s (2018, p. 15) five reuse principles: “Try to *reuse ... as a whole*, if that is not possible as parts ...” as discussed in Section 2.5.5.

The crux is that reuse is not a formal prerequisite for generating a new software application. Instead, the desire to reuse grows over time as a project progresses. When teams start considering smaller software pieces for reuse, over time, these turn out to be hard to locate if specifications are to be met and difficult to integrate. Soon it is too late in the process to follow KPMG’s (2018) related reuse principle.

Cases 1–4 demonstrate that *internal reuse within the IT provider* gives a good starting position for systematic reuse. It allows for internal and, therefore, direct client feedback, offers immediate and often reliable funding for the asset owning team and increases credibility and expertise during a client demo.

Reuse can be maximised by maximising the number of reusers within one deal (Case 3), the number of reuses of the solution for the same client (Case 1), vertically within the same domain, and horizontally across the domain as indicated in Cases 1 and 2. Every time the abstraction level increases, so does the effort for adaptation. The reuse cases are logically derived from each other by abstraction.

Reuse takes place in different experimental situations, as described in Section 2.3. As discussed in Case 1, abstraction helps find new, related reuse situations. Consequently, *a logical, abstract connection exists between the reuse situations and the use cases of a*

reusable software asset. In the example of the luggage handling software mentioned in Section 1.3, the logical connection was routing items: the material flow.

Every further *adaptation* has an impact on the existing reuser community. Changes need to be decided upon so that a maximum number of reusers will benefit while keeping the effort required to a minimum (C2-I4).

Current reuse reporting focuses on the business benefit (Case 3). However, *sharing (re)use case details* could help with the abstraction required to brainstorm new reuse cases for the asset.

According to Cases 4 and 5, mapping *the software asset prerequisites against opportunity lists makes sense*. This action can be performed by the solutioning team (C4-I3). Case 4 demonstrates the most systematic reuse.

According to Case 2, systematic reuse will be achieved if documentation and the solution are standardised. Reusing one software asset instead of providing many individual one-off solutions has a *standardising effect* on the business.

Security matters. The client has to stick to his security requirements. According to Cases 1-3 and 5, software asset reuse should be in sync with security requirements on the client side.



Software asset reuse aims at a B2B niche market. While reused software is of higher quality and value-proofing the shift to a value-creating circular economy (Van Buren et al., 2016), the reuse of valuable software assets is not ubiquitously applicable but limited to a *niche business in B2B*. Software products serve the primary market.

The overall finance-driven objective of creating a reusable software asset is to generate revenue (Birch, 2017). However, interviewees pointed out an efficiency-driven objective: “to stop the never-ending *parade of unfunded assets* being developed within accounts” that generate excess cost while bringing only a subset of the functionality of the jointly developed software asset (C3-I1, L35), as mentioned in Section 4.4.1. This confirms Wilson’s (2000) valuation of the software asset concept (see Figure 2.4), while Birch’s (2017) purely revenue-focussed asset concept should be revisited.

Answering Jonsson’s (2021) what is reused: This case study shows software assets can be reused throughout the entire life cycle of a project. The researcher noticed two roles

an asset can play – content asset or methodology asset (see Table 4.3). *Content assets*, as seen in Cases 1 and 2, containing knowledge related to the business content of a project, can be proposed for reuse during the project’s sales cycle. Since they add content to the client’s business process, the researcher calls them “content assets”. They frequently constitute the primary reason leading to the closure of a contract. These assets improve the client’s efficiency, resulting in revenue, or the internal efficiency of the IT provider, resulting in savings, and sometimes in both.

Table 4.3 Content asset versus methodology asset (source: author)

Reuse initiation situation Company-owned asset ...	Seller initiated reuse	Solutioning team initiated reuse	Developer/delivery initiated reuse
Generating revenue when reused 	Asset 2 for Case 2 Content Assets	Asset 3 for Case 3 Methodology Assets	n/a
... generating internal savings 	Asset 1 for Case 1	Asset 4 for Case 4	Asset 5 for Case 5

In contrast, some reusable assets, e.g., Cases 3-5, aim at methodologies like testing or roll-out. They optimise processes and methodologies and can be referred to as *methodology assets*. They “stop the never-ending parade of unfunded assets being developed” all for the same purpose (C3-I1, L35). They are less likely to get proposed at an early sales stage. Usually, they are considered during solutioning or even after contract closure. They are not for independent sale (C3-I2). The asset owning team uses the content asset to support the account team in fulfilling its business content mission. The methodology asset acts like a tool or accelerator. Some assets, e.g., Asset 2 and 3, can be either reused in the role of content or methodology asset.

Software asset reuse is a discipline. IT management cannot expect this to be done by reuse novices who are paid to code. As software products have a marketing team, reusable software asset teams need experts. So, for example, *a reuse manager* could be put in place – a paid professional accountable for hands-on managing reuse supporting multiple asset owning teams in parallel to build up and leverage reuse experience.

Figure 4.42 summarises the benefits of systematic software asset reuse resulting both from literature and case study research. The reuse advantages newly identified by Cases

1–5 are highlighted in bold. They demonstrate how this research enriched the current theory of knowledge work in the area of knowledge reuse.

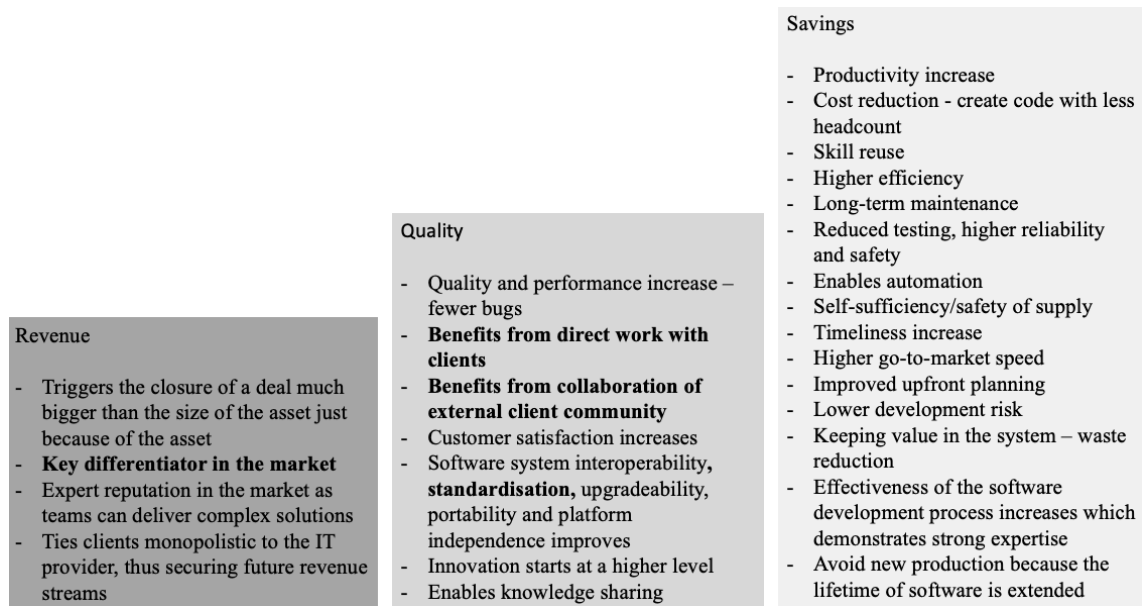


Figure 4.42 Benefits of software asset reuse – source (Cases 1–5 – bold entries) and (Barros-Justo et al., 2018; Horne, 1995; Majchrzak et al., 2004; Mihale-Wilson et al., 2021; Ram & Devi, 2019; Salgot et al., 2017; Sandhu & Batth, 2021b; Wilson et al., 2000)

Figure 4.43 summarises the above-discussed asset reuse-related case study findings generalisable across all five cases.

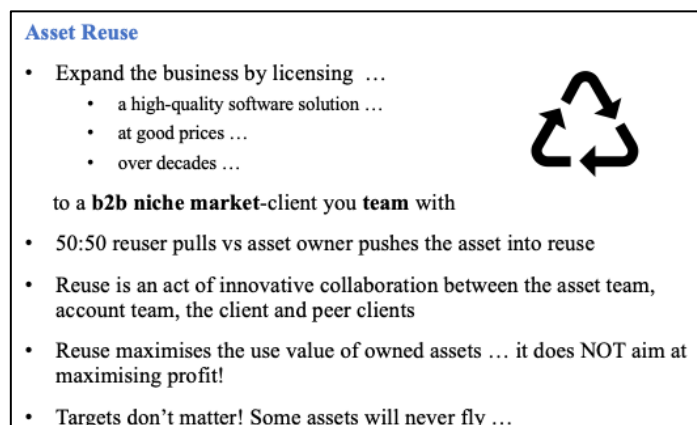


Figure 4.43 Summary of case study outcome – asset reuse (source: author)

For overview purposes, Figure 4.44 sums up this case study's major, most interesting, generalisable findings. The figure reflects the just discussed four sections – Asset Creation, Section 4.7.1; Asset Coding/Storage, Section 4.7.2; Asset Distribution, Section

4.7.3; and Asset Reuse, Section 4.7.4, respectively. The blue arrows in Figure 4.44 indicate the logical sequence – first, an asset must be created, then coded and stored, then distributed and only then reused. This reuse itself sometimes triggers the creation of a new asset.

Case Study Findings

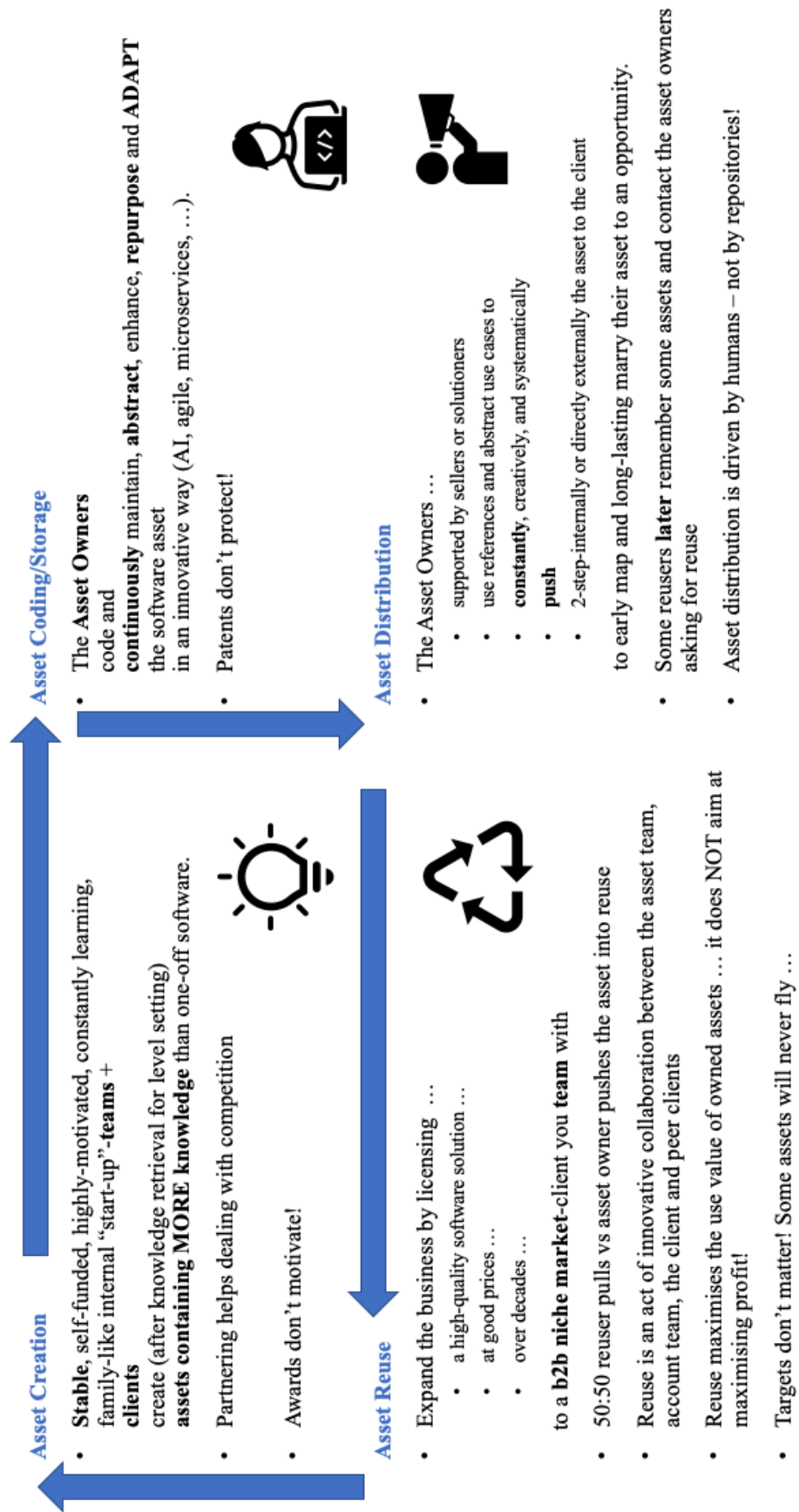


Figure 4.44 Summary of the key case study findings (source: author)

4.8 Answers to the Research Questions

This thesis presents a case study examining the critical knowledge-related processes required for achieving systematic software asset reuse in a global IT company. Following Rasche & Seisreiner (2018), a four-stage approach was applied. First, the addressed problem was researched by looking at the literature, feeding the findings into the semi-structured interview guide and running semi-structured interviews to gather practical insights from experts in the field. Second, several alternatives were identified during the literature and case study research. Third, these alternatives were analysed and discussed. Fourth, the research questions' answers reflect this analysis's outcome.

RQ1: How can a reusable software asset be characterised?

Zhou's (2001) and Spoelstra's (2011) explanation of a reusable software asset presented in Section 2.1 needs to be expanded to include:

A reusable software asset is *valuable, intangible, non-rival, IT provider-owned, identifiable, unique, adaptable* intellectual property comprising *software* and *related IP artefacts*. It aims to shape a current or *future* IT service in a *niche* market. A reusable software asset can be licensed to B2B clients incurring licence or service fees. Alternatively, it can be reused for the advantage of the B2B client or internally, generating efficiency savings and other *benefits*. Complex, solution-like, reusable software assets result from the ongoing *collaboration* of the asset-owning team, account team, client and group of peer clients.

Figure 4.45 provides the underlying logic behind this answer. The reusable software asset inherits all characteristics of the classes it belongs to – following the literature research in Chapter 2. The lefthand side of Figure 4.45 lists how a reusable software asset can be classified. On the right side in grey, Figure 4.45 maps the resulting characteristics of a reusable software asset.

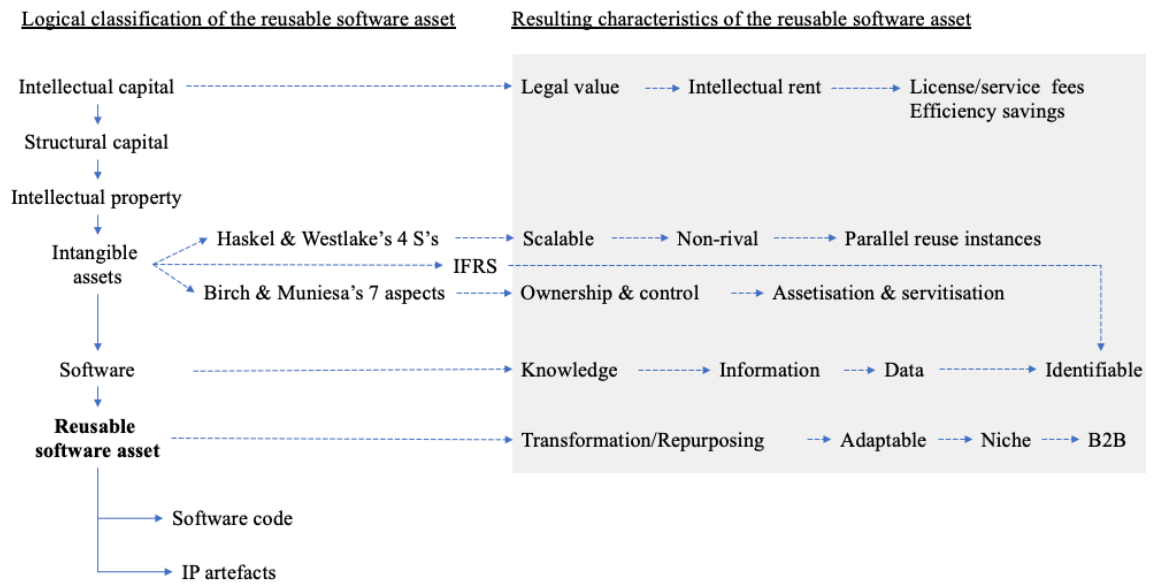


Figure 4.45 Characteristics of the reusable software asset (source: author)

A reusable software asset represents a *legal value* in financial reporting since it represents intellectual capital (Horne, 1995). Because of its targeted but uncertain future benefits, a reusable software asset's legal value represents a discounted, expected return on investment that is difficult to assign until it is realised via intellectual rents. Reusing a software asset can result in efficiency *benefits* internally and externally. Licence or service revenue generation requires the reuse decision to be taken prior to closure of the client contract – as in Cases 2 and 3 of the presented case study. Throughout its reuse, the reusable software asset remains the property of the IT provider. The reuser can obtain the right, e.g., via a licence, to use it without owning it.

Traditional one-off software, scenario I in Figure 2.16, results from a request from the client and is owned by the client. Even if subcontracted, the final control of the software is with the client (Table 4.4, row I). In contrast, the IT provider retains *ownership* and control (Atasu et al., 2021; Birch & Muniesa, 2020) of the reusable software asset and the product throughout their lifetime to generate income via licence fees or at least savings. This ownership includes responsibility for waste and risk costs (Press room European Commission, 2014). Control determines, e.g., the availability of updates. Since clients and their peers co-shape the reusable software asset, they can co-influence the control of the IT provider. For details, see rows II and III in Table 4.4, which correspond to scenarios II and III in Figure 2.16 respectively.

Table 4.4 Comparing the use of conventional one-off software, reusable software asset-and software product (source: author)

Use of	Usership	Ownership	Control (e.g. updates)	Business	Created by
Conventional One-off Software (I)	Client / (IT)	Client	Client	B2B	Client/IT
Reusable Software Asset (II)	Client/(IT)	IT	(Client)/IT	B2B	Client Client-Peers IT-Asset-Owner IT-Account-Team
Software Product (III)	Client	IT	IT	B2B/B2C	IT

Given that software is *intangible*, a reusable software asset is *non-rival* (Haskel & Westlake, 2018), as explained in Section 2.5.1. Non-rival means that the consumption of the software does not reduce the available amount. Thus it can be used with one client and simultaneously reused with other clients in parallel, as explained in Section 2.2.2 (Jones & Tonetti, 2020) – a fact evident in all cases of the case study. It adds complexity and differentiates the reuse of intangibles from the reuse of tangibles, where the reuse situation remains limited to one. Contrasting Haskel & Westlake (2018) and Romer (1990), in this case study, nonrivalry neither leads to price-taking nor nearly unbounded monopolistic competition (Romer, 1990). One reason could be a fast-emerging spillover. The key inputs in reusable software assets require clever ideas – so reuse depends on human capital (Alrowwad & Abualoush, 2020). Expanding Romer’s (1990) findings, it can be concluded that a larger total stock of intellectual capital (Abd-Elrahman & Ahmed Kamal, 2020; Azzahra, 2018; Flores et al., 2020), as discussed in Section 2.2.1, will lead to faster economic growth.

A software asset must be *adaptable* to fit a client’s individual business situation. Each reuse transforms a source asset into a final asset (Lear, 2021), as detailed in Section 2.5.5. On-premises assets show a behaviour where, typically, each client has a different version of the software asset. It undergoes many small releases. If an IT provider productises his asset by keeping one asset version for all clients, as in the cloud instance in Case 2, all the clients receive a multitude of micro-releases. This individual adaptation allows *niche* businesses where software products do not fit to be served. The fact that each client receives a slightly different, *unique* IT service indicated by the yellow star and green

triangle differentiates the reusable software asset at the top of Figure 4.46 from the software product, shown at the bottom of Figure 4.46.

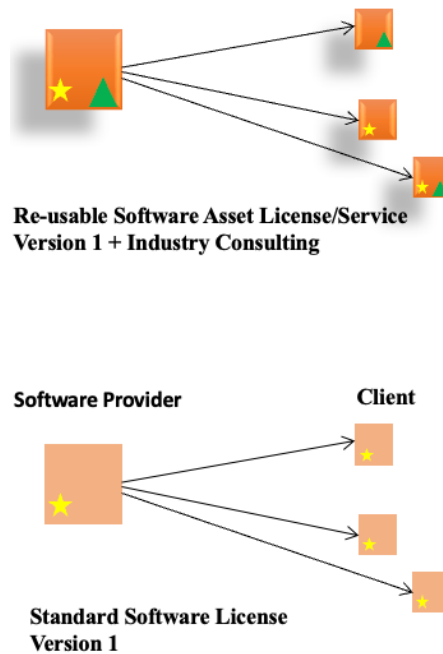


Figure 4.46 Reusable software asset versus software product - adapted from (Wolfram et al., 2020)

Enhancing the asset is an *ongoing process* to improve reusability and maximise reuse. It results from the close relationship between the asset owner, account team, and client and peer clients, aiming to nurse and grow their asset “baby” jointly (CI-II, L474). The asset owners in Cases 1–4 share features created upon one client's request with all other clients (see the purple rhombus in Figure 4.47). Created upon the request of Client A, it is incorporated into the asset and later shared with Clients C and D. This joint asset adaptation activity further differentiates a reusable software asset from a software product.

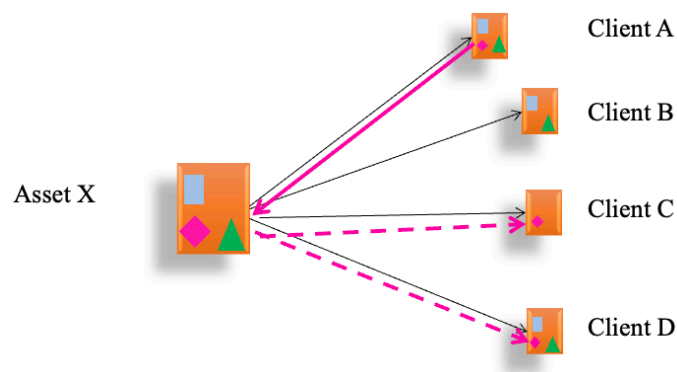


Figure 4.47 Peer clients co-shape the reusable software asset (source: author)

Software is *intangible*. As shown in Figure 4.48, software, in general, can be a software asset for business IT. There are several kinds of software assets, e.g., the reusable software asset or the software product.

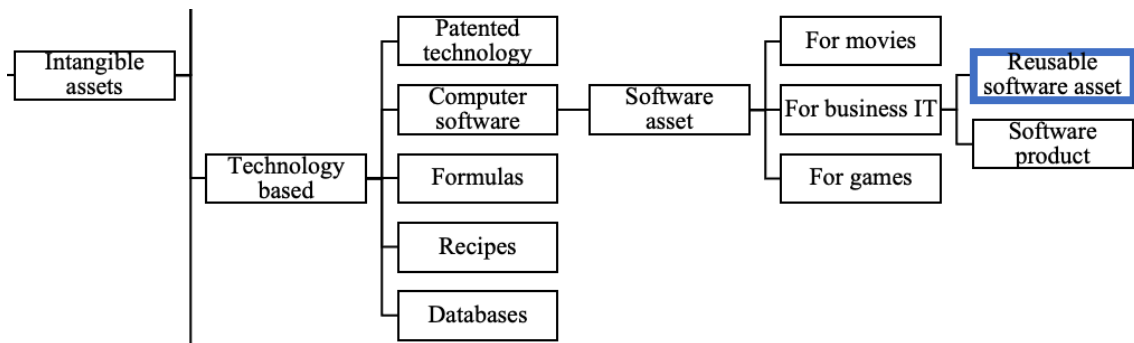


Figure 4.48 The reusable software asset is a specific software asset - adapted from (Mehta & Madhani, 2008)

A reusable software asset consists of *software and related IP artefacts*. The IP artefacts can comprise explicit, documented, sharable and reusable knowledge, information and data such as related test cases, functional designs, code components, test scenarios, objects, design models, domain architecture, database schema, documentation, manuals, standards, offerings, solutions, or sales material. Technical rules that help make these artefacts reusable include modularisation, independence, robustness, standardisation, adaptability, extendibility, portability, or maintainability (Krueger, 1992; Reddy & Jalender, 2011; Sandhu & Batth, 2021a).

The adaptations made to the reusable software asset at each reuse have a refreshing effect. Consequently, reusable software assets have a longer lifetime (Lear, 2021) than one-off software. Therefore, reusable software assets shape not only the *current but also the future IT service*. They are a bet on the future. Subsequently, today's asset could be reused later for a job it may never have been designed for.

The knowledge contained in the asset typically increases with heightened *collaboration*. Consequently, a reusable software asset contains more knowledge and fewer errors than an individual one-off solution. It is, therefore, of higher quality due to the evolutionary innovation (Wegener, 2016).

Reusable software assets are aimed at *B2B* clients, see the green box in Figure 4.49. If there were a market bigger than just a niche market for this software, the form of a software product would have been chosen (see left side of Figure 4.49). Deciding on an individually adaptable reusable software asset (see right side of Figure 4.49) allows market niches to be addressed attractively. The effort of marrying the software asset to an opportunity can only be taken in B2B situations where the reuse charges are higher – not for B2C. With the introduction of the cloud, this could change. On an individual contract basis, B2B reuse fees are typically higher than those of B2C software products. This justifies the effort required for client interaction, manual adaptation and finding a new client.

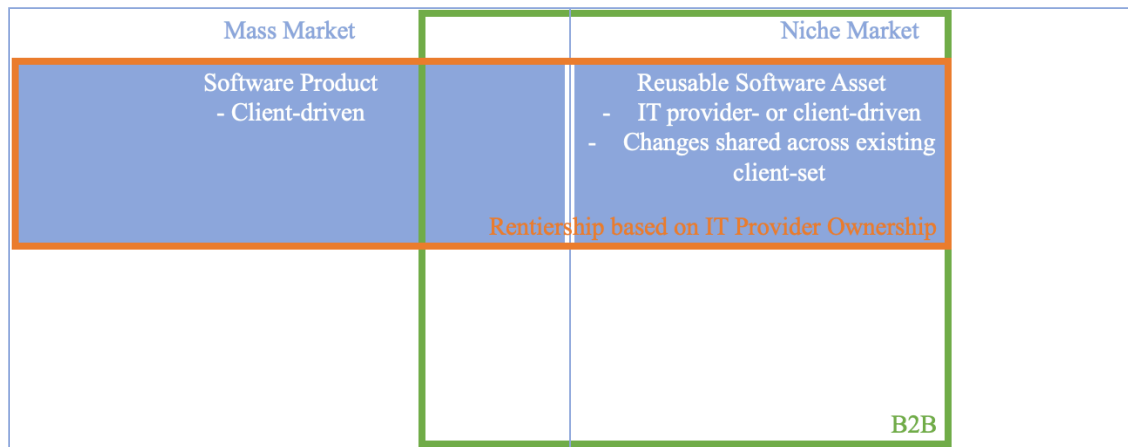


Figure 4.49 The reusable software asset positioned against the software product (source: author)

RQ2: How can the systematic reuse of a software asset in a global IT enterprise be specified?

Griss' (1996) definition of systematic software asset reuse presented in Section 2.1 needs to be expanded to include the following:

Information about the existence of the asset has to be externalised and *distributed* across space and time to attract new clients. Each reuse exposes the existing reusable software asset to *a different experimental system* – at the risk of initially not having all the required functions in place. Horizontal reuse requires more abstraction to repurpose the existing software asset than vertical reuse creatively. Technical *adaptation* takes manual intervention to *innovatively* close the gap between the given and the software asset's future function set for each client reuse. Systematic reuse is an *organisational capability*. It gives the ad-hoc distribution effort methodological and strategic guidance to increase the number of potential reuses over time. Reuse is *collaborative*. It is triggered by and compensates for *scarcity* as it maintains assets and increases the *benefits* of their reuse.

Following are the explanations for why the amended definition is needed.

Internal or external marketing helps *distribute* the news on the existence of the reusable software asset from the asset owner asset across time and space to potential reusers brainstorming a new experimental system.

As detailed in Section 2.3, bundling expertise into a reusable software asset is an evolutionary *innovation* that shapes the ground for revolutionary innovation (Hargadon, 2002; Wegener, 2016), like repurposing.

Repurposing (C1-I3) is the innovative act of identifying a specific business opportunity as the new experimental system for the asset, e.g., sorting parcels at DHL with the help of luggage-routing software. A good overview of the latest use cases eases *abstraction* to identify new use cases and new client segments in new market niches for the reusable software asset via brainstorming, mind mapping or other creativity methods. Abstraction provides the virtual link between all reuses across the different experimental systems, disregarding the individual item functions.

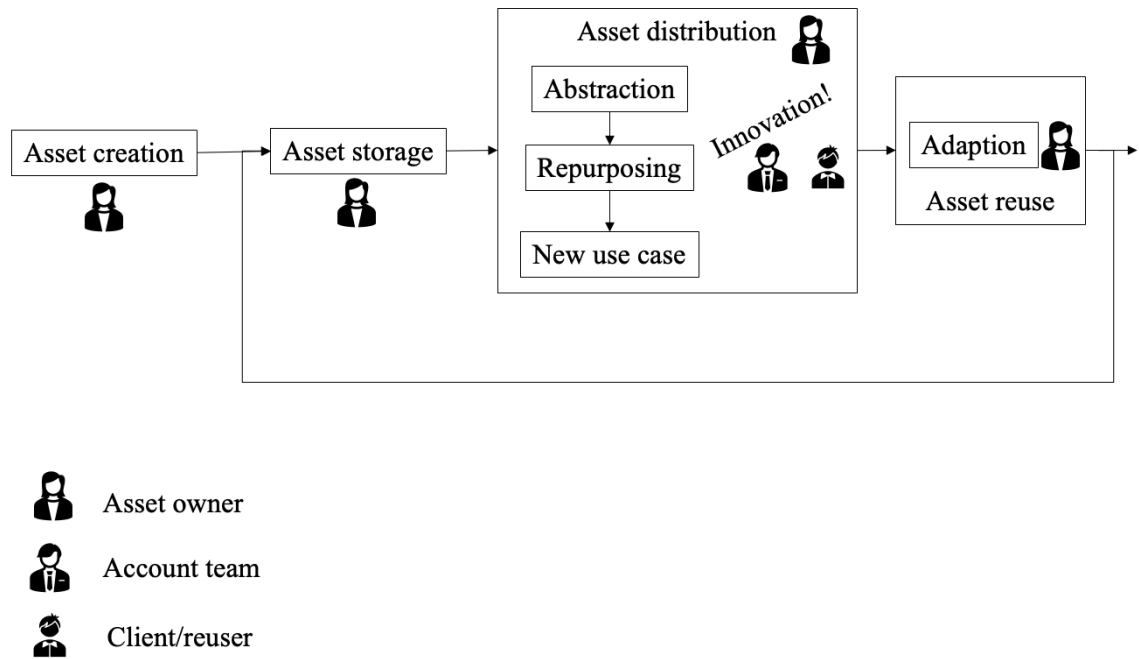


Figure 4.50 Software asset reuse process (source: author)

Apart from creation and storage, reuse depends on distribution across space and time and here in particular on abstraction, repurposing for a new use case and subsequent adaption as Figure 4.50 indicates. This forms the missing puzzle piece in Figure 2.22.

Systematic reuse influences the listed factors so that reuse can be increased. In practice, matching each asset against each opportunity during each primary project stage can be inefficient. Clustering assets (blue cubes in Figure 4.51) in families by industry, use cases, offering or solution groups allows a controlled, traceable asset-to-project matching process (Gall, 2004), see blue arrows in Figure 4.51, to set in at the earliest point in time, see wedding rings at the start of the blue project P2 in Figure 4.51. Optimising the pre-agreed and planned process over time by prioritising strategic matches characterises the *organisational capability* and maximises reuse benefits.

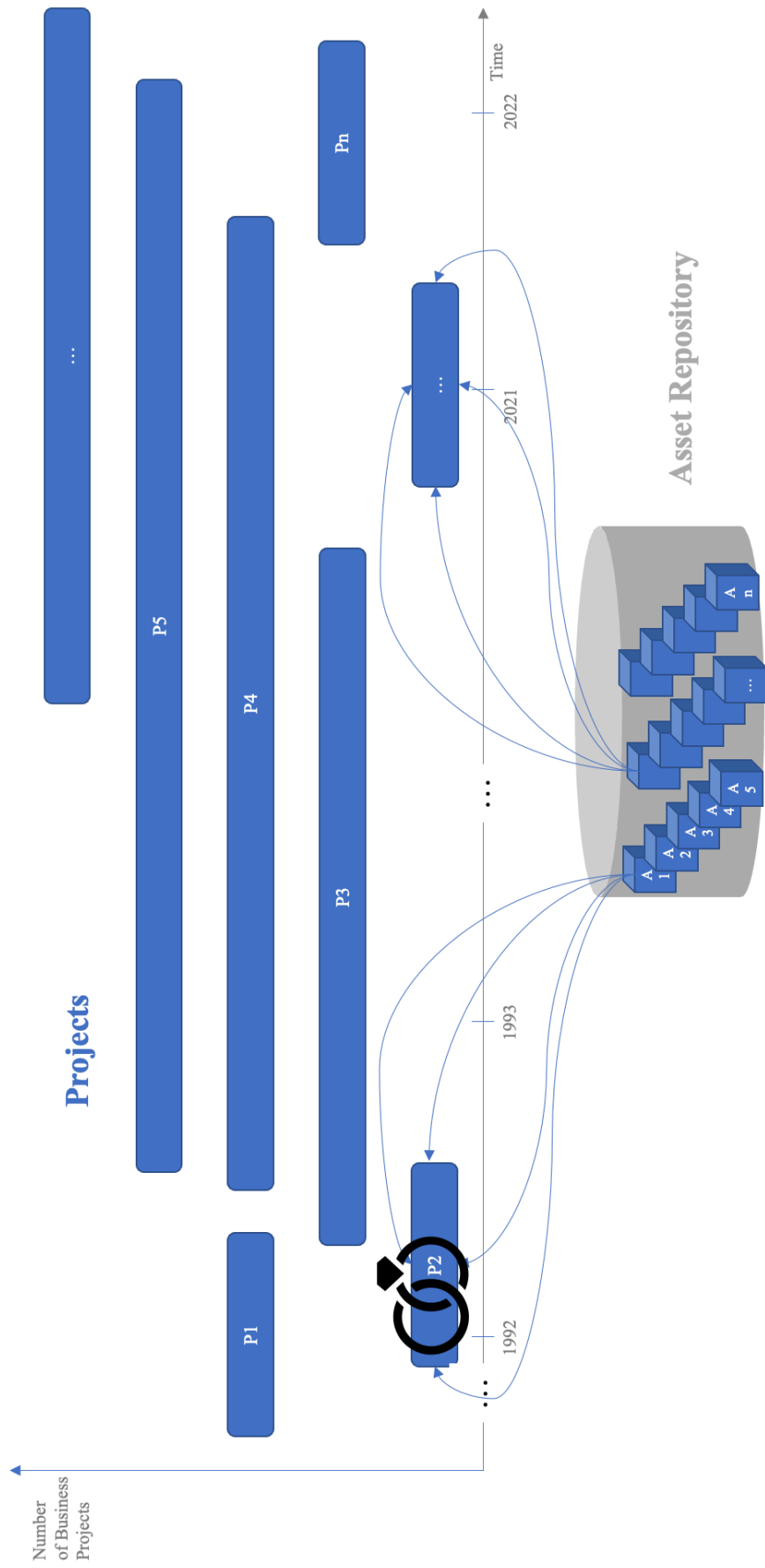


Figure 4.51 Marrying a reusable software asset to an IT business project (source: author)

Table 4.5 A reusable software asset versus software product - adapted from (Cacioppo et al., 2015; Schloss, 2018)

Item functions	Same experimental system	Different experimental system
Same item functions	Reproducibility	Replicability
	Software product	
Different item functions / variants	Robustness	Generalisability/Reuse
		Reusable software asset

As shown in Table 4.5, reuse can only occur if the reusable software asset exposed to a *different experimental system* is being adapted. The resulting different item functions are the software features.

Every asset adaptation (Rose, 2019; Van Buren et al., 2016) impacts future reuses (Case 2). Therefore, decisions during software *adaptations* regarding technology, flexibility, openness and partnerships should be taken carefully, with a view to future reuses that require future adaptations. Some adaptations of the asset are always required. Even after as many reuses as in Case 4, a currency change from US dollars to Swedish krona was required. Technological advances drive additional changes.

Any modification to the software asset should be small. Minor adaptations have a lower cost and maintain the focus of the asset in one area. They are typical for vertical software reuse. However, sometimes new client segments must be targeted. Extensive modifications enabling horizontal reuse cost more and require strategic decisions (C2-I3) on where the asset will go, as discussed in Section 4.3.4. The newly developed features move the reusable software asset closer to or further from the next set of clients that can be targeted.

Reuse allows a precious resource with limited availability temporarily and locally to be better managed. As discussed in Section 2.4.8, “reuse is an important part of successful design in general” (Jonsson et al., 2021, p. 1). Its success depends on three factors: at what point in time the search for reusables begins, the reuse reasons and conditions, and what finally gets reused (Jonsson et al., 2021).

This research has demonstrated that the right time to start reusing a software asset is when *scarcity* is perceived, as explained in Section 2.3.1. It takes an honest assessment to state, e.g., the lack of sufficient knowledge when complexity sets in.

RQ3: How should the knowledge-related processes critical to enabling systematic software asset reuse in a global IT enterprise be shaped?

The case study indicates that systematic software reuse goes beyond reminding teams to match the right software assets with the opportunity in focus. Instead, systematic software asset reuse requires a holistic, carefully chosen approach for software asset creation, storage, distribution and reuse. Continuous improvement can refine this approach over time.

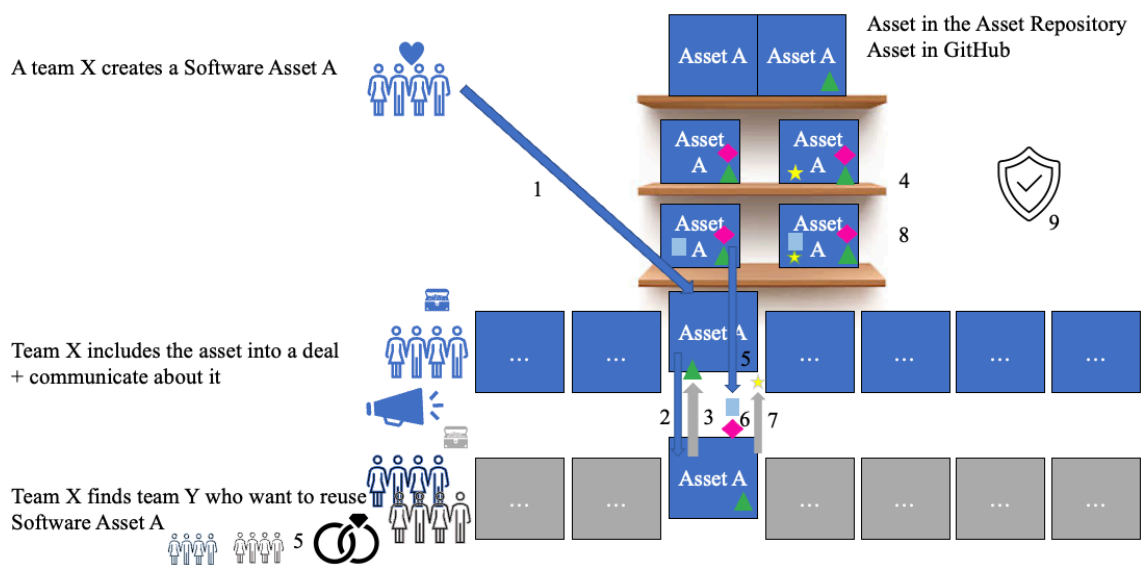


Figure 4.52 Software asset reuse according to Cases 1–5 of the Case Study (source: author)

Figure 4.52, summarising how the software asset reuse took place for Cases 1–5, contrasts with Figure 2.20 in the literature review section. The asset owning team creates the software asset (1) for an initial project marked in blue. It then pushes the same asset into a second opportunity marked in grey (2). The reusable software asset (indicated by the little green triangle) must be adapted to make it work. This adaptation results in a helpful feature and is, therefore, shared with the initial project marked in amber (3). Now, further projects are onboarded, which leads to new asset features indicated in pink and light blue (4), which are passed on to the existing projects in blue and grey (6).

Further changes in the grey project led to another new feature (yellow star) (7). All these changes are reflected in the different asset versions (8). Security concerns (9) have to be addressed in parallel.

By putting the literature and case study findings into perspective, practical managerial guidelines for systematic software asset reuse can be derived, addressing Garcia's (2007) call for clear top-down reusability support discussed in Section 2.1. They result from the abductive research approach (see Section 3.2.1) and provide the practical implications of the findings of this research.

Managerial guide on how to enable systematic software asset reuse

Asset Creation

The goal of creating an asset should be carefully considered, as reuse is for *value preservation* – not profit maximisation based on throughput maximisation (Stahel & MacArthur, 2019).

Internal and external *market analysis* should be run before creating a software asset to identify already existing software in this area. A SWOT analysis could provide strategic guidance. The new software asset could either bundle skills on the IT provider side to provide unique services to clients or bring efficiency savings by preventing a never-ending parade of small one-off software (Case 3). A reusable software asset typically has an extended lifetime (Lear, 2021) and higher value – as it accumulates the knowledge of more experts over a longer time – compared to a one-off software sold to the client. The market niche the asset addresses should be in line with the IT provider's market strategy.

Reusable software is an *investment into the future*. The market analysis will provide input for the funding decision. It often helps to have a permanent internal client for the reusable software asset as this provides a constant source of funding for the asset owning team – giving them time to find external reuse clients. The asset owning team should start lightly to hold their ground for longer life in the market.

Software creation is best done by a competent, highly skilled team working like a “family” business (C1-I3) in a constant team setup, forming a centre of competence in an atmosphere of trust and collaboration shaped by their visionary communicative leadership (Dong et al., 2015; Koloniari et al., 2019; Teerajetgul & Charoenngam, 2006; Ueki et al.,

2011), as discussed in Section 2.4.5. High employee job motivation could counteract inertia and the NIH syndrome inherent in all humans. A decision should be made upfront about what centre of competence this asset team will represent, how it fits into the strategy of the IT provider and how the asset team could be leveraged as a *human capital asset*.

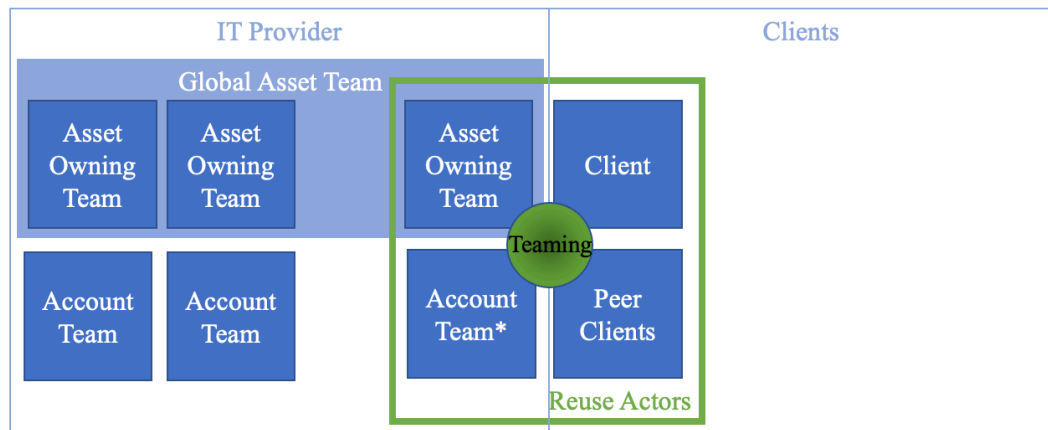
The asset owning team should *factor in change* when creating a new reusable software asset. It is in the nature of reuse that due to each new reuse situation; the reusable software asset needs to be adapted (Cacioppo et al., 2015; Schloss, 2018). So, the reusable software asset should be set up with change in mind. Frozen black-box-software-constructs should be avoided; their reuse is highly unlikely.

These changes, done in interaction with the client, limit software asset reuse to *B2B only*. They would not be affordable B2C. Now, within B2B, reusable software assets only serve *niche* clients since software products can do all mass business. As discussed in Section 2.5.5, reuse can answer the growing number of *niche* markets (Jarvis, 2009).

Asset Storage

The asset owners remain in touch throughout the entire reuse period with their software asset reusers, as discussed in Section 4.2.2. The reusable software asset absorbs the knowledge of the wider client set. After receiving one client's request, the software asset is typically *adapted*, and then it is offered as a code extension to the existing set of ongoing clients. This way, all clients benefit from each other. This aligns with Jarvis' (2009) comment in Section 2.5.4 on Amazon's deep knowledge of its clients. It requires controlled asset management and client management.

A reusable software asset collaboratively and innovatively integrates ideas from a set of reuse actors - clients, the asset owning team, the account team and indirectly the peer clients. These individual teams represent wider groups of experts – the IT provider, the global asset team, or the set clients - as shown in Figure 4.53. The time and effort for *collaboration* need to be factored in.



* Including Solutioning Team

Figure 4.53 Teams involved in software asset reuse (source: author)

Software storage requires high-quality knowledge content fed as input into a high-quality knowledge management system that provides a high-quality knowledge management service. This increases user satisfaction and intensifies use (Taraszewski, 2017). Software asset storage requires abstraction for classification (Sandhu & Bath, 2021a) and (C1-I4). It is, therefore, helpful to focus on use cases. In addition, it is recommended that all source codes for all versions should be stored and kept for later reference (Di Cosmo & Zacchiroli, 2017) and (C1-I2).

The key to systematic software asset reuse is to understand the basic purpose of the reusable software asset. Knowledge management allows the *repurposing* and reuse of the software asset via *abstraction*. If the current purpose of the asset lacks sufficient business opportunities, it is necessary to focus on the prerequisites and continually modify and adapt the software to broaden its applicability.

The reuse savings are counter-weighted by *constantly maintaining the asset* until the next business opportunity is won.

Successful assets can be reused over a long time (C1-I2), during which the IT, the business and specific client requirements can change. The architecture should be designed with *decades of change* and yet unspecified future reuse in mind.

Asset Distribution

It cannot be expected that clients will request a reusable software asset as they would request a software product. The reusable software asset is too specific to bring it to the attention of thousands of clients. Centrally developing, evaluating and communicating methods for *software asset distribution* can represent a new, additional role for the global asset organisation of the IT provider.

Figure 4.54 summarises the identified distribution options of reusable software assets presented and discussed in Sections 4.2 - 4.7. Information on the asset can either be pushed out by the asset owning team (at the right in Figure 4.54) or pulled in by the reusing team (at the left in Figure 4.54).

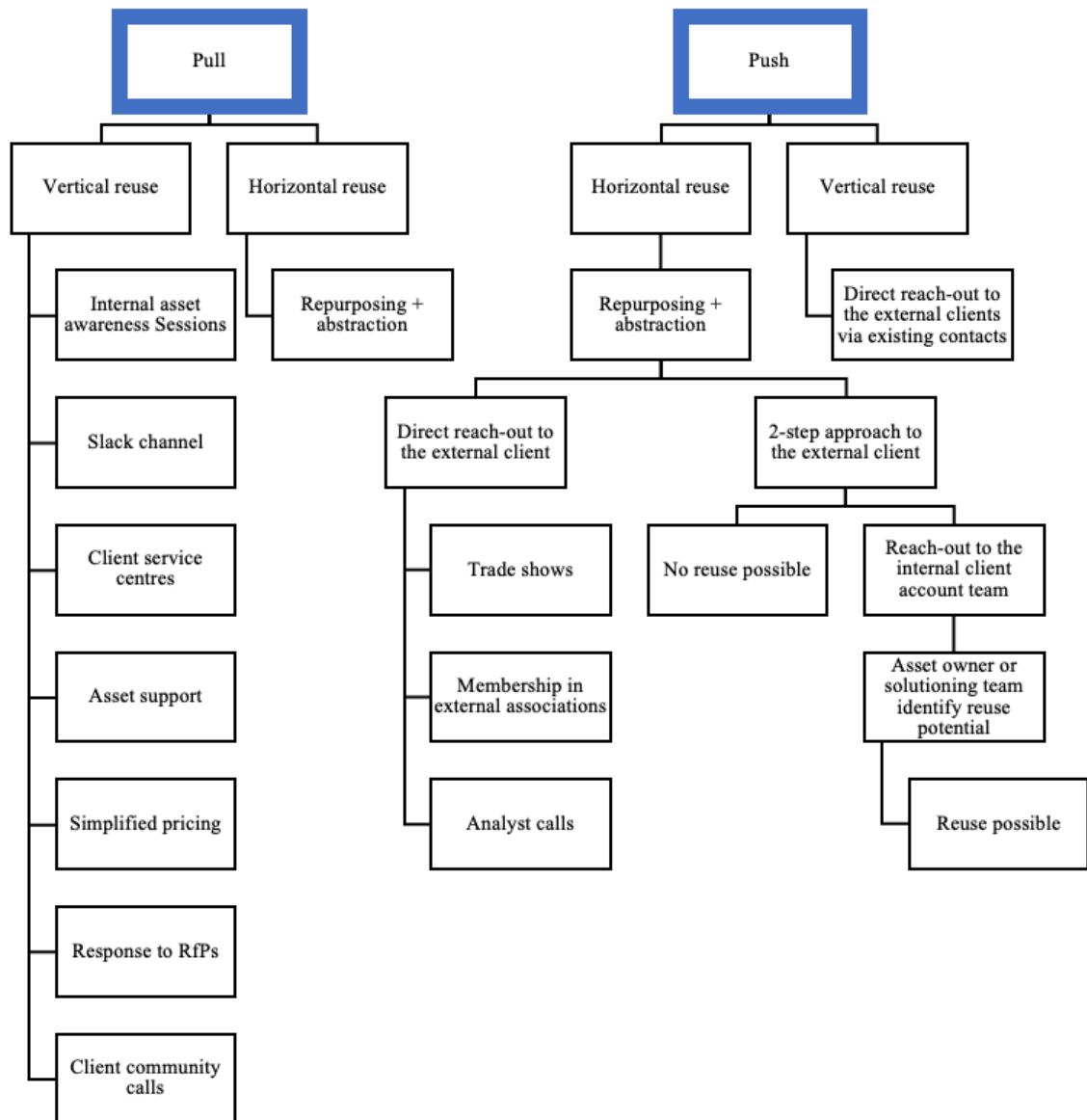


Figure 4.54 Overview of Marketing Methods for Reusable Software Assets (source: author)

Asset marketing is pivotal for the success of a reusable software asset because circa half of all reuse cases are initiated by teams that do not own the reusable software asset and have only heard about it. The message needs to be targeted to the audience, e.g., software sellers need different input than developers.

Horizontal reuse requires a *two-step reuse approach*. First, the asset owning team has to contact and convince the account team. Then the external client can be contacted.

For effective systematic reuse analysis through record scanning, the asset owning team must comprehensively understand the *technical and non-technical asset requirements*. For example, if the reuse of Case 5 on test automation were a good fit for opportunities employing more than five testers, it would make sense to search for related projects actively.

Software assets are primarily documented internally. In a world where employees and clients google everything, as described in Section 2.5.4 by Jarvis (2009), *software features should be searchable on the internet* for people and machines in the most detailed way. This could increase the chance of locating a reusable software asset.

Asset Reuse

Amongst all closed IT deals, many are extensions. Further, many opportunities are initiated by a client in the form of a tender or RfP leaving little space for the reuse of narrow-shaped reusable software assets. Subsequently, *reuse situations are limited*.

Software asset reuse is not standard; it is a process that rarely happens and requires extra attention. *Triggers* could help reinforce the reuse process so that reuse is considered early and more often rather than just by coincidence.

Increasing the number of valid *matches between software assets and business projects* and gaining higher reliability in matching could provide helpful reusability metrics. Matching is a job that the management should assign to a team of solutioning experts who continuously document and refine their asset-to-project matching skills, and their success should be measured based on this. Here the global asset organisation could provide strategic guidance. It takes conscious management-supported effort to implement reuse, focusing on long-term system change (Van Buren et al., 2016).

Systematic reuse is an ability an organisation must acquire by developing related reuse capabilities and achieving maturity in these (Succar, 2013). Sharing software assets requires management to create an *atmosphere of trust and a sharing culture* that provides employees with autonomous job motivation (Abdelwhab Ali et al., 2019; Akosile & Olatokun, 2020; Hejase et al., 2014).

Pushing a few strategic or commercially significant, well-documented, well-maintained software assets to be matched with projects based on their use case may be more efficient than waiting in vain for practitioners to voluntarily pull a reusable software asset out of a crowd of undocumented and unmaintained assets (Wallace, 2018).

Reuse criteria for each asset should be explicitly stated to avoid frustration among those interested in reusing the asset. Further, the asset owning team should track why reuse is not going to happen – often because another prerequisite for reuse has been identified. Knowing all the prerequisites of reuse helps to focus asset marketing. The references, use cases, and excluding factors should be made available in the asset repository. Over time it may be good to *remove reuse obstacles* during subsequent development.

Repositories containing meta-information about reusable software assets are important but should not be overestimated. As indicated in Section 2.4.8, successful reusers tend to *personally contact the asset owner* (Terjesen, 2003). It may, therefore, make sense to install and promote a figurehead per asset, someone potential reusers know and remember.

Figure 4.55 provides a flowchart on the software asset reuse tasks. The step increasing reuse comprises the tasks detailed in Figure 4.50. To close the loop, feedback in the form of reuse reporting is required, as discussed in Section 2.4.8.

The teams highlighted in Figure 4.53 profit from software asset reuse. *Clients* benefit from the collaboration that goes along with software asset reuse. They should be open to teaming with peer clients and understand the enormous value reuse provides.

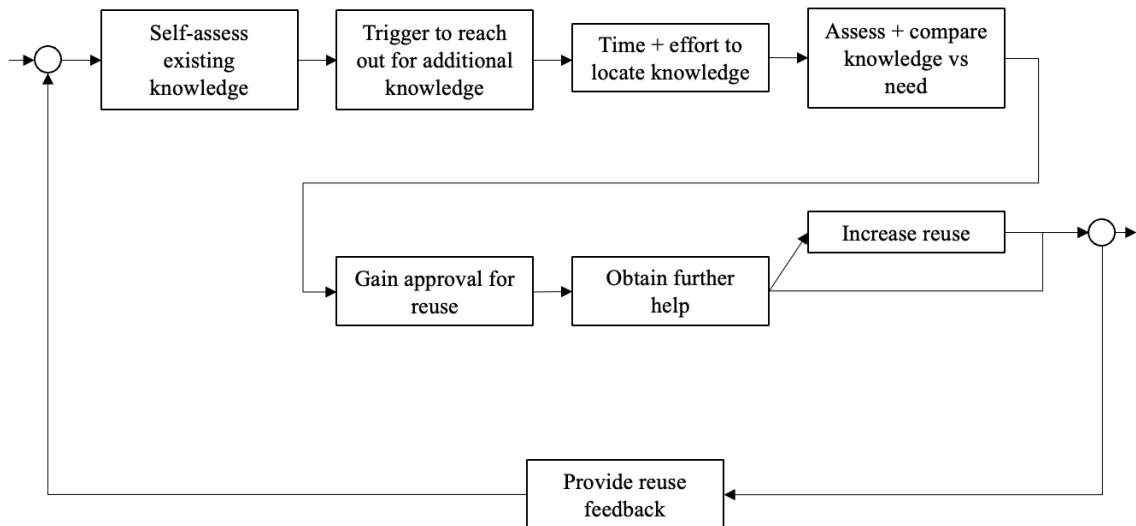


Figure 4.55 Software asset reuse tasks (source: author)

The *account team* ought to consciously identify the need for reuse and pull reusable software assets into their business opportunities. Once they have identified a suitable software asset, they need to enable the collaboration between their client and the asset owning team to adapt the asset to the market requirements and benefit from further changes to come. In addition, the account team must take the first step towards creating a brand-new reusable software asset when it makes sense.

The *asset owning team* must actively seek new reuse opportunities for their software asset. They orchestrate asset change and collaboration.

The *global asset team* could act as a reuse knowledge multiplier by virtually connecting existing asset owning teams to share knowledge on asset marketing actions. Further, the global asset team should provide and maintain the technical infrastructure required for software asset reuse, such as providing an asset-sharing platform.

The *managers* of the IT provider ought to take a long-term business decision on where and to what extent the IT provider should invest in software asset reuse.

Figure 4.56 summarises the practical implications of this study per each expert group, i.e. client, account team, solutioning team, asset owning team, global asset team and management team of the asset owning company. Highlighted in bold are new practical insights this research gave.

Practical Implications of Software Asset Reuse

Client	Account Team	Solutioning Team	Asset Owning Team	Global Asset Team	Management
<p>Beneficiary + Co- Collaborator</p> <ul style="list-style-type: none"> Be open to reuse - understand the full value Software Asset Reuse provides Be an active member of the asset improving community by virtually teaming with peer clients 	<p>Connector by pulling asset into project + Birth parent of new asset</p> <ul style="list-style-type: none"> Be open to reuse - understand the full value Software Asset Reuse provides <ul style="list-style-type: none"> Reuse gives you experts Reuse gives you proven software - NOT old rubbish Team with the Asset Owning Team and your client in promoting and improving the asset Pull the asset into your deal (50%) Consciously decide to create a new reusable software asset if required 	<p>Connector by pulling into project</p> <ul style="list-style-type: none"> Suggests assets for reuse Bridge between execution and presales 	<p>Building assets + connector by pushing asset out to other teams</p> <ul style="list-style-type: none"> Be part of the active and supportive Global Asset Team Identify, understand and share all asset use cases at a level that allows horizontal reuse as a result of abstraction Communicate the exact asset's prerequisites with potential reusers along with your will to adapt them Understand the client's situation - adapting the prerequisites increases reusability Push the asset into the market (50%) Asset launch advisor Harvesting lessons learned on the use of the asset 	<p>Efficiency driver</p> <ul style="list-style-type: none"> Leverage the creation, storage, distribution and reuse actions across all the asset owning teams to increase efficiency and effectiveness → Strive for systematic software asset reuse Work towards making the reuse of an identified asset as easy as possible, as fast as possible and as early as possible (low threshold) Refuse to take black-box assets on board – if they cannot be adapted they cannot be reused! Advocate reuse to the management Provide and maintain the technical infrastructure required for reuse Drive reuse efficiency 	<p>Investor</p> <ul style="list-style-type: none"> Addresses purely B2B niche areas Software Asset Reuse is a long-term investment – it cannot be short-term switched on or off Asset Owning Teams are centres of competence Asset Owning Teams should be staffed with people who think long-term and do not aim at the next career step Software Asset Reuse taps a niche in the IT market – an extra area of business Software Asset Reuse can provide a constant source of income Software Asset Reuse counteracts local or temporary scarcity of knowledge or resources It is in its nature that Software Asset Reuse will not maximise profit

Figure 4.56 Practical implications of software asset reuse (source: author)

5. Contributions and Conclusion

This Chapter summarises the contribution to knowledge (Section 5.1) and practice (Section 5.2) this thesis makes in targeting the three identified research gaps in a way IT providers can practically benefit from this research. Section 5.3 lists the research limitations. Finally, Chapter 5 offers recommendations for future research (Section 5.4) and a definitive conclusion to promote reuse in IT and beyond (Section 5.5).

5.1 Contribution to Knowledge

This thesis provides academic contributions in the research domains of *intellectual capital*:

Utilising theory such as Abeyssekera's (2021, p. 1) conceptualisation of "knowledge-based value creation" in conjunction with the International Financial Reporting Standards' (IFRS, 2022) framework on software licensing onto the case study contributes to a more nuanced understanding of the *distinctions between reusable software assets and software products*. Addressing a gap in the existing literature, this thesis identifies by offering tailored solutions to a heterogeneous client base while retaining intellectual property rights, reusable software assets cater for *a dedicated business need*. The theoretical implication is to broaden the scope of the IFRS (2022) by explicitly differentiating between these two business-relevant types of software assets, thereby enhancing the clarity and applicability of the financial reporting standards in the context of software asset categorisation.

Further, incorporating the theory of the International Financial Reporting Standards' (IFRS, 2022) perspective on the recognised business necessity for reusable software assets yields the theoretical inference that the *applicability of software asset reuse is predominantly confined to business-to-business niche markets*.

The theory of Lacy et al. (2020) on how reuse expands the life of tangible items has the theoretical implication that it, even more, applies to intangibles due to the in this thesis identified setting of *parallel intangible instances of reuse extending the longevity of the reusable items*. As a side effect, these parallel instances foster fresh business opportunities.

Drawing upon the theoretical frameworks of Romer (1990) and Haskel & Westlake (2018), which suggest that assets can lead to market concentration and dominance, as well as Birch & Muniesa's (2020) assertion that assets frequently entail forms of monopolistic control, the case study contributes to knowledge by researching cases where the spillover effect typically prevents monopolistic competition when intangibles such as software assets are reused. As a theoretical impact, this confirms Haskel & Westlake's (2018) indication. Further, this research expands Birch's (2020) definition of assetisation to non-monopolies and emphasises the importance of *servitisation* and *assetisation* in facilitating the reuse of intangible assets.

Emphasising the relationships between the above-mentioned intellectual capital-related theories, it can be seen Abeysekera's (2021) conceptualisation of knowledge-based value creation and the International Financial Reporting Standards' (IFRS, 2022) framework on software licensing provide a foundation for understanding the differences between reusable software assets and software products. The integration of these theories allows for the expansion and clarification of the IFRS (2022) regarding software asset categorisation. Moreover, the theories of Romer (1990), Haskel & Westlake (2018), and Birch & Muniesa (2020) highlight the potential for market concentration and dominance while also examining cases where intangible asset reuse prevents monopolistic competition.

This thesis adds to the academic literature in the *circular economy* research domain:

In response to the concerns articulated by Keswani (2014) and Brown (2021) regarding the limited presence of *real-life software reuse examples* in scholarly literature, this case study presents, for the first time, an exploration of intangible reuse instances characterised by shared usership while preserving ownership. The theoretical implication, in particular emerging from the first-time description of parallel reuse instances, posits that *reuse constitutes a legitimate business approach* enhancing the sustainability-focused circular economy literature by Stahel (2019) and others.

Utilising Griss' (1996) theoretical framework, the fundamental attributes of a reusable software asset can be elucidated. The case study's theoretical implications necessitate a

more expansive definition that incorporates the aspect of distribution of reusable software and related artifacts, thereby facilitating large-scale intangible reuse.

In accordance with Wallace' (2018) theory, assets need to be pushed for reuse, which can either be horizontal or vertical (Anasuodei & Ojekudo, 2021; Jalender et al., 2010). The theoretical implications derived from this case study research extend the notions of vertical and horizontal reuse to the circular economy research domain and contribute to the prevailing theory by highlighting the necessity of a two-step sales/distribution process, particularly in the context of horizontal reuse.

Outlining how the above-mentioned circular economy theories are interrelated and provide a better understanding of the research contribution it can be summarised that this thesis addresses concerns raised by Keswani (2014) and Brown (2021) about the scarcity of real-life software reuse examples. By incorporating Griss' (1996) theoretical framework, the study offers a more comprehensive definition of reusable software assets and their distribution. Furthermore, Wallace's (2018) theory on asset reuse and the concepts of vertical and horizontal reuse (Anasuodei & Ojekudo, 2021; Jalender et al., 2010) are extended to the circular economy research domain, emphasising the importance of a two-step sales/distribution process.

This research constitutes a valuable addition to the scholarly discourse within the realm of *knowledge management*:

In alignment with Fogg's (2009) theory, both motivation and ability are essential preconditions; however, a trigger (Dreyer & Wynn, 2016) is required to prompt a behavioural shift from usage to reuse, as exemplified by scarcity (Salgot et al., 2017). The ensuing theoretical implication posits that *the reuse of reusable software assets is triggered by knowledge scarcity* in space or time while decoupling reuse management from the technical aspect of IT.

By integrating Lacy's (2020) servitisation theory and Fayyaz's (2020) knowledge-sharing theory with the insights gleaned from this case study, the theoretical implications suggest that *sharing asset adaptations crafted for one client among the existing client base* fosters a feedback loop, thereby endowing the reusable software asset with a quality surpassing that of a conventional software product.

Incorporating Wegener's (2016) creativity and knowledge brokering theory, along with Sandhu & Batth's (2021a) abstraction theory, into this research yields the theoretical implication that *abstraction plays a crucial role in both knowledge generation and the virtual connection of distinct knowledge instances*.

Drawing upon Alrowwad & Abualoush's (2020) theory, according to which *intellectual capital is, via reuse, turned into innovation* and business performance, the theoretical implication emerges that *innovation can be realised through the abstraction and repurposing of knowledge assets within specialised domains*. This confirms and enriches the theoretical framework in Figure 2.22.

Applying the theory of Wijnhoven (2008), Alavi & Leidner (2001) and others, this thesis bridges a research gap by *detailing the knowledge management processes towards systematic software asset reuse*. This has several theoretical implications. One not yet listed is, the existing literature on knowledge creation, see Section 2.4.5, needs to be enhanced by the step of *checking for existing knowledge for level setting* and potential reuse of an already existing reusable software assets prior to creating a new asset. Further, asset creation has an element of uncertainty. A reusable software asset is a *bet on the future* - today's asset could be reused later for a job it may never have been designed for.

Emphasising the relationship between the above-mentioned knowledge management-related theories, it can be summarised that Fogg's Field (2009) theory on motivation and ability, coupled with Dreyer & Wynn's (Field (2016) concept of triggers, highlights the role of knowledge scarcity in prompting behavioural shifts towards reuse. The integration of Lacy's (2020) servitisation theory and Fayyaz's (2020) knowledge-sharing theory suggests that sharing adapted reusable software assets can create a feedback loop, enhancing the quality of these assets. Additionally, Wegener's (2016) creativity and knowledge brokering theory, Sandhu & Batth's (2021a) abstraction theory, and Alrowwad & Abualoush's (2020) theory on intellectual capital and innovation all contribute to understanding the importance of abstraction in knowledge generation and the repurposing of knowledge assets within specialised domains.

5.2 Contribution to Practice

This thesis contributes to the industry by providing an *actionable managerial guide* that facilitates collaborative change based on the following newly identified software asset reuse principles.

The *compelling reasons for reusing a software asset* are to expand business or increase efficiency by stopping the “never ending parade of unfunded assets being developed” (C3-I1, L35).

It takes a stable, competent asset owning team acting as a *family-run start-up-like business team* to create and maintain a reusable software asset and protect it against internal and external competition.

The long-term reuse of a software asset is complex and, therefore, requires centralised management by the *asset owner* - it cannot be spread over multiple reusers. This aspect impacts the transition of IT services from a current to a new provider.

Software asset reuse can benefit from a dedicated *reuse manager* ideally responsible for hands-on management of reusable software assets, supporting multiple asset-owning teams simultaneously, and leveraging reuse experience to enhance software asset reuse as a discipline.

The team of asset owners and their clients form a *virtual community collaboratively improving the reusable software asset* over time.

Valuable software assets that cannot be *adapted* cannot be reused.

Reuse is not a mass business - to date, it requires *manual intervention* to handle each reuse situation specifically.

5.3 Limitations

In Section 3.2.2, this work abandoned action research in favour of case study research. The small number of only five cases makes generalisations challenging. Action research could have better identified causality and factors influencing software asset reuse. However, despite these limitations, case study research remains valuable, given careful consideration to rigour and effectiveness. Action research has its own limitations, including the risk of researcher bias.

This research explores how the knowledge required to reuse software assets can be managed to benefit asset-owning IT organisations. Since “the adoption of new technology has been efficient only if the culture of the organisation is in sync with the technology requirements” (Upadhyay & Kumar, 2020, p. 9), *organisational culture is a factor* that has not been looked at as part of this study. Similarly, other factors outside knowledge management could impact the systematic reuse of software assets.

This case study *was limited to a few months only*. Indeed, more insight could have been gained by doing longitudinal research over decades.

The *researcher may be biased* due to work experience, which could affect the study’s legitimate outcome. On the other hand, her work experience allowed her to understand better what the interviewees said – given company language and abbreviations.

To bolster the robustness of the derived model and validate the research theory, it would be advantageous to *verify the research results across cases beyond the IT provider and industries* mentioned in Section 1.4. This broader validation would enrich the model’s applicability.

This research focuses on the reuse implications from the *perspective of one multi-national IT company* and how they can further improve their business. To overcome this limitation, further studies could be conducted on multiple companies, local companies or teams purely operating onsite.

Possible ways to overcome potential limitations in subsequent studies would be to alter research factors such as methodology, sampling, timing, or the research team.

5.4 Recommendations for Future Research

Case study results can provide input for generalisation (Polit & Beck, 2010). Because abductive reasoning allows only for an incomplete experience, further research taking a community approach could harden the findings of this work or go beyond. Here are some recommendations for application in the industry.

Moving towards a resource-based view, see Table 2.2, will expand the lifetime of items by preserving value and minimising waste (Fivet & Brütting, 2020; Lear, 2021). Reuse as a concept, and its *social-economic effects* (Birch & Muniesa, 2020) could be further explored.

This thesis focuses on reusing *entire software applications* – not on reusing single lines of code. To overcome this limiting factor, it would be interesting to investigate how reuse findings change along the size of the reusables.

This thesis has worked out that *intangibles can be reused in a B2B niche market* if they can be adapted to the new experimental situation. Future research could demonstrate if the generalised findings of this research can be leveraged in the IT industry and beyond. Examples to test could be the reuse of data (Custers & Uršič, 2016; Pasquetto et al., 2019), a brand (Hasnain & Mohseni, 2018), goodwill, customer relationships, efficient business processes in civil engineering or pharmacy or the reuse of intangible design assets in the automotive industry leading to reuse of related tangible production assets.

Future research could investigate the *internal analytics required to match existing assets with business opportunities*. Upadhyay & Kumar (2020) perceive the analytics knowledge required as tacit.

Reuse as part of the circular economy has been studied mainly in terms of tangible assets (Stahel & MacArthur, 2019). Given that intangible assets differ in their characteristics from tangible assets and since the importance of intangible assets is increasing in today's world (Haskel & Westlake, 2018, 2022), additional effort should be put into *researching the circular economy, including reuse, based on a wider set of intangible assets*.

Researching the underlying abstraction in reuse cases and how this relates to different experimental systems, the reuse situations, could help identify future reuse situations faster and more successfully. It would be helpful to understand how AI can support here.

Following Jarvis' (2009) comment on the importance of *sharing new software features so they can be googled online*, a study could be conducted to evaluate its effects on locating reusable software assets. More work should be undertaken on improving the trustworthiness of and information about reusables.

While the present study centred on the business facets of software asset reuse, the potential *implications of emerging IT trends such as cloud computing and generative AI* remain intriguing avenues for further exploration.

The discerned *interrelation between software asset reuse and the circular economy* presents a promising lever for fostering environmental preservation and resource conservation, meriting in-depth investigation and scrutiny.

5.5 Conclusion

The reuse of complex software assets is evident in practice. Following a hype in the past, scientific research withdrew prematurely from the topic.

A reusable software asset constitutes an intangible, non-rival, unique, adaptable intellectual property which is routed in the knowledge embodied within software and related IP artefacts. By reusing, i.e. adapting this asset, it can be applied in specialised fields, particularly in niche business-to-business markets where standardised software products are non-existent, and individual client necessitate tailored solutions.

This requires the knowledge owner engages in knowledge creation with reusability in mind, ensures knowledge storage over an extended period, and facilitates knowledge distribution to permeate novel application areas.

The knowledge reuser must consciously recognise his knowledge scarcity and be willing adopt the pre-existing knowledge of others and subsequently contribute further knowledge throughout the adaptation process. Software asset reuse offers broader benefits than previously thought.

Implementing software asset reuse is not purely an IT topic but has a business aspect that connects intellectual capital, circular economy and knowledge management research.

A reusable software asset represents a long-term business investment. Organisational capability should facilitate perpetual collaborative software changes, which are pivotal for systematic software asset reuse.

References

- Abd-Elrahman, A.-E. H., & Ahmed Kamal, J. M. (2020). Relational capital, service quality and organizational performance in the Egyptian telecommunication sector. *International journal of emerging markets*, 17(1), 299-324. <https://doi.org/10.1108/IJOEM-11-2019-0983>
- Abdelwhab Ali, A., Panneer selvam, D. D. D., Paris, L., & Gunasekaran, A. (2019). Key factors influencing knowledge sharing practices and its relationship with organizational performance within the oil and gas industry. *Journal of knowledge management.*, 23(9), 1806-1837. <https://doi.org/10.1108/JKM-06-2018-0394>
- Abdulameer, Z. A., & Sati'Abbas, S. (2020). Adaptive reuse as an approach to sustainability. *IOP Conference Series: Materials Science and Engineering*, 881(1), 1-16. <https://doi.org/10.1088/1757-899X/881/1/012010>
- Abeyssekera, I. (2021). Intellectual capital and knowledge management research towards value creation. From the past to the future. *Journal of Risk and Financial Management*, 14(6), 238. <https://doi.org/10.3390/jrfm14060238>
- Abualoush, S., Masa'deh, R., Bataineh, K., & Alrowwad, A. (2018). The role of knowledge management process and intellectual capital as intermediary variables between knowledge management infrastructure and organization performance. *Interdisciplinary Journal of Information, Knowledge, and Management*, 13, 279-309. <https://doi.org/10.28945/4088>
- Ackoff, R. L. (1989). From data to wisdom. *Journal of applied systems analysis*, 16(1), 3-9.
- Agostini, L., & Nosella, A. (2017). Enhancing radical innovation performance through intellectual capital components. *Journal of intellectual capital.*, 18(4), 789-806. <https://doi.org/10.1108/JIC-10-2016-0103>
- Ahmed, S. S., Khan, M. M., Khan, E., Sohail, F., & Mahmood, N. (2021). Enhancing intellectual capital and organizational performance through talent management. In *The Dynamics of Intellectual Capital in Current Era* (pp. 205-220). Springer Singapore. https://doi.org/10.1007/978-981-16-1692-1_10
- Ahrne, G., & Papakostas, A. (2001). *Inertia and innovation* (S. University, Ed. Vol. 5). SCORE (Stockholm Center for Organizational Research).
- Airport-Industry-Review-cms. (2011, 2011-04-13). *Behind baggage handling*. Airport Technology. Retrieved 2022-05-28 from <https://www.airport-technology.com/features/feature116081/>
- Aithal, P. S. (2017). An Effective Method of Developing Business Case Studies Based on Company Analysis. *International Journal of Engineering Research and Modern Education (IJERME)*, 2(1), 16-27. <https://doi.org/http://doi.org/10.5281/ZENODO.400579>
- Akhmad, M., Chang, S., & Deguchi, H. (2021). Closed-mindedness and insulation in groupthink: their effects and the devil's advocacy as a preventive measure. *Journal of Computational Social Science*, 4(2), 455-478. <https://doi.org/10.1007/s42001-020-00083-8>

- Akosile, A., & Olatokun, W. (2020). Factors influencing knowledge sharing among academics in Bowen University, Nigeria. *Journal of Librarianship and Information Science*, 52(2), 410-427. <https://doi.org/10.1177/0961000618820926>
- AL-Badareen, A. (2021). Reuse alternatives based on the sources of software assets. *International Journal of Computer and Information Technology* (2279-0764), 10(1), 18-24. <https://doi.org/10.24203/ijcit.v10i1.67>
- Alavi, M., & Leidner, D. E. (2001). Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS quarterly*, 25(1 (Mar., 2001)), 107-136. <https://doi.org/https://doi.org/10.2307/3250961>
- Ali, I., Musawir, A. U., & Ali, M. (2018). Impact of knowledge sharing and absorptive capacity on project performance: The moderating role of social processes. *Journal of Knowledge Management*, 22(2), 453-477. <https://doi.org/10.1108/JKM-10-2016-0449>
- Alrowwad, A. a., & Abualoush, S. H. (2020). Innovation and intellectual capital as intermediary variables among transformational leadership, transactional leadership, and organizational performance. *Journal of Management Development*, 39(20), 196-222. <https://doi.org/10.1108/JMD-02-2019-0062>
- Ampatzoglou, A., Bibi, S., Chatzigeorgiou, A., Avgeriou, P., & Stamelos, I. (2018). Reusability index: A measure for assessing software assets reusability. In R. Capilla, et al. (Ed.), *New Opportunities for Software Reuse* (Vol. 10826, pp. 43-58). Springer International Publishing. https://doi.org/10.1007/978-3-319-90421-4_3
- Amschler Andrews, A., & Pradhan, A. S. (2001). Ethical issues in empirical software engineering: The limits of policy. *Empirical Software Engineering*, 6(2), 105-110. <https://doi.org/10.1023/a:1011442319273>
- Anasuodei, M., & Ojekudo, N. A. (2021). Software reusability: approaches and challenges. *International Journal of Research and Innovation in Applied Science (IJRIAS)*, VI(V), 142-146. <https://doi.org/10.51584/IJRIAS.2021.6510>
- Anderson, W., Morris, E., Smith, D., & Ward, M. C. (2007). *COTS and reusable software management planning: A template for life-cycle management* (Acquisition support program, Dynamic systems program, Issue. S. E. Institute.
- Antons, D., Declerck, M., Diener, K., Koch, I., & Piller, F. T. (2017). Assessing the not-invented-here syndrome: Development and validation of implicit and explicit measurements. *Journal of Organizational Behavior*, 38(8), 1227-1245. <https://doi.org/10.1002/job.2199>
- Archibald, M. M., Ambagtsheer, R. C., Casey, M. G., & Lawless, M. (2019). Using Zoom videoconferencing for qualitative data collection: Perceptions and experiences of researchers and Participants. *International Journal of Qualitative Methods*, 18(1-8), 2-8. <https://doi.org/10.1177/1609406919874596>
- Arvanitou, E.-M., Ampatzoglou, A., Chatzigeorgiou, A., & Carver, J. C. (2021). Software engineering practices for scientific software development: A

- systematic mapping study. *Journal of Systems and Software*, 172, 1-33. <https://doi.org/10.1016/j.jss.2020.110848>
- Aspers, P., & Corte, U. (2019). What is Qualitative in Qualitative Research. *Qualitative Sociology*, 42(2), 139-160. <https://doi.org/10.1007/s11133-019-9413-7>
- Atasu, A., Dumas, C., & van Wassenhove, L. N. (2021). The Circular Business Model. *Harvard Business Review*(July - August 2021), 1-14.
- Atasu, A., & Van Wassehove, L. (2021, 2021-11-10). *The circular economy: From enthusiasm to realism*. INSEAD - The Business School for the World. Retrieved 2022-09-03 from <https://knowledge.insead.edu/responsibility/the-circular-economy-from-enthusiasm-to-realism-17656>
- Atieno, O. P. (2009). An analysis of the strengths and limitation of qualitative and quantitative research paradigms. *Problems of Education in the 21st Century*, 13, 13-18.
- Auerbach, P. (2016). Education as a Social Process. In *Socialist Optimism* (pp. 191-224). Palgrave Macmillan UK. https://doi.org/10.1007/978-1-137-56396-5_8
- Azzahra, K. (2018). The influence of human capital, structural capital and relational capital to the performance of cooperation with competitive advantage as intervening variable of cooperation in South Tangerang. *EAJ (Economic and Accounting Journal)*, 1(1), 24-34.
- Baetjer, J., Howard. (2000). Capital as embodied knowledge: Some implications for the theory of economic growth. *The Review of Austrian Economics*, 13(2), 147-174. <https://doi.org/10.1023/a:1007808618703>
- Baillon-Bachoc, N., Caron, E., Chevalier, A., & Vion, A.-L. (2021). Providing software asset management compliance in green deployment algorithm. In S. M. Thampi, E. Gelenbe, M. Atiquzzaman, V. Chaudhary, & K.-C. Li (Eds.), *Advances in Computing and Network Communications - Proceedings of CoCoNet 2020* (Vol. 1, pp. 451-464). Springer Singapore. https://doi.org/10.1007/978-981-33-6987-0_37
- Bangen, K. J., Meeks, T. W., & Jeste, D. V. (2013). Defining and assessing wisdom: A review of the literature. *The American Journal of Geriatric Psychiatry*, 21(12), 1254-1266. <https://doi.org/10.1016/j.jagp.2012.11.020>
- Baptista, A., Frick, L., Holley, K., Remmik, M., & Tesch, J. (2015). The doctorate as an original contribution to knowledge: Considering relationships between originality, creativity, and innovation. *Frontline Learning Research*, 3(3), 55-67. <https://doi.org/10.14786/flr.v3i3.147>
- Barney, J. B. (2001). Resource-based theories of competitive advantage: A ten-year retrospective on the resource-based view. *Journal of management.*, 27(6), 643-650. <https://doi.org/10.1177/014920630102700602>
- Barr, P. N. (2013). *Does Skype help or hinder communication*. Retrieved 2022-09-03 from <https://theconversation.com/does-skype-help-or-hinder-communication-21121>

- Barros-Justo, J. L., Benitti, F. B. V., & Matalonga, S. (2019). Trends in software reuse research: A tertiary study. *Computer Standards & Interfaces*, 66, 1-18. <https://doi.org/10.1016/j.csi.2019.04.011>
- Barros-Justo, J. L., Pincioli, F., Matalonga, S., & Martínez-Araujo, N. (2018). What software reuse benefits have been transferred to the industry? A systematic mapping study. *Information and Software Technology*, 103, 1-21. <https://doi.org/10.1016/j.infsof.2018.06.003>
- Bauer, V., & Vetro, A. (2016). Comparing reuse practices in two large software-producing companies. *Journal of Systems and Software*, 117, 545-582. <https://doi.org/10.1016/j.jss.2016.03.067>
- Bauer, V. M. (2016). *Analysing and supporting software reuse in practice* [Technische Universität München]. München. <https://dblp.org/rec/phd/dnb/Bauer16a>
- Beha, F., Schmalstich, M., Brüning, M., Disilvestro, M., & Makama, T. (2022). *Sharing economy - business model toolbox*. Retrieved 2022-09-03 from <https://bmtoolbox.net/patterns/sharing-economy/>
- Behutiye, W., Rodríguez, P., Oivo, M., Aaramaa, S., Partanen, J., & Abhervé, A. (2022). Towards optimal quality requirement documentation in agile software development: A multiple case study. *Journal of Systems and Software*, 183, 1-17. <https://doi.org/10.1016/j.jss.2021.111112>
- Bernard, V., Schultheiss, S. J., & Michaut, M. (2014). Learn from the best. *PLoS Comput Biol*, 10(5), e1003645. <https://doi.org/10.1371/journal.pcbi.1003645>
- Bidee, J., Vantilborgh, T., Pepermans, R., Huybrechts, G., Willems, J., Jegers, M., & Hofmans, J. (2013). Autonomous motivation stimulates volunteers' work effort: A self-determination theory approach to volunteerism. *VOLUNTAS: International Journal of Voluntary and Nonprofit Organizations*, 24(1), 32-47. <https://doi.org/10.1007/s11266-012-9269-x>
- Biggerstaff, T. J. (1994). The library scaling problem and the limits of concrete component reuse. Proceedings of 1994 3rd International Conference on Software Reuse, Rio de Janeiro, Brazil.
- Birch, K. (2017). Rethinking Value in the Bio-economy. *Science, Technology, & Human Values*, 42(3), 460-490. <https://doi.org/10.1177/0162243916661633>
- Birch, K., & Muniesa, F. (2020). *Assetization: Turning things into assets in technoscientific capitalism* (K. Birch & F. Muniesa, Eds.). The MIT Press. <https://doi.org/10.7551/mitpress/12075.001.0001>
- Birch, K., & Tyfield, D. (2013). Theorizing the bioeconomy: biovalue, biocapital, bioeconomics or... what? *Science, Technology, & Human Values*, 38(3), 299-327.
- Borge, M., Ong, Y. S., & Rosé, C. P. (2018). Learning to monitor and regulate collective thinking processes. *International Journal of Computer-Supported Collaborative Learning*, 13(1), 61-92. <https://doi.org/10.1007/s11412-018-9270-5>

- Bouthillier, F., & Shearer, K. (2002). Understanding knowledge management and information management: the need for an empirical perspective. *Information research*, 8(1), 1-39.
- Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2), 77-101. <https://doi.org/10.1191/1478088706qp063oa>
- Brown, P., Von Daniels, C., Bocken, N., & Balkenende, A. (2021). A process model for collaboration in circular oriented innovation. *Journal of Cleaner Production*, 286, 1-18. <https://doi.org/https://doi.org/10.1016/j.jclepro.2020.125499>
- Brydon-Miller, M., Greenwood, D., & Maguire, P. (2003). Why Action Research? *Action Research*, 1(1), 9-28. <https://doi.org/10.1177/14767503030011002>
- Burciu, A., & Kicsi, R. (2015). Knowledge as a distinctive resource of competitive advantage. *Ecoforum Journal*, 4(1), 9-14.
- Cacioppo, J. T., Kaplan, R. M., Krosnick, J. A., Olds, J. L., & Dean, H. (2015). *Social, behavioral, and economic sciences perspectives on robust and reliable science* (Report of the Subcommittee on Replicability in Science Advisory Committee to the National Science Foundation Directorate for Social, Behavioral, and Economic Sciences, Issue. <http://web.stanford.edu/group/bps/cgi-bin/wordpress/wp-content/uploads/2015/09/NSF-Robust-Research-Workshop-Report.pdf>
- Carayannis, E. G. (2012). Absorptive capacity and organizational learning. In N. M. Seel (Ed.), *Encyclopedia of the sciences of learning* (pp. 25-27). Springer, Boston, MA. https://doi.org/10.1007/978-1-4419-1428-6_1620
- Chappell, T. (2012). Varieties of knowledge in Plato and Aristotle. *Topoi*, 31(2), 175-190. <https://doi.org/10.1007/s11245-012-9125-z>
- Clarkson, G., Jacobsen, T. E., & Batcheller, A. L. (2007). Information asymmetry and information sharing. *Government Information Quarterly*, 24(4), 827-839. <https://doi.org/10.1016/j.giq.2007.08.001>
- Coghlan, D. (2019). *Doing action research in your own organization* (A. Owen, Ed. 5th ed.). SAGE Publications Ltd.
- Cohen, W. M., & Levinthal, D. A. (1990). Absorptive capacity: A new perspective on learning and innovation. *Administrative science quarterly*, 35, 128-152. <https://doi.org/10.2307/2393553>
- Conford, F. M. (1935). The Collected Dialogues of Plato [Translation]. *The collected dialogues of Plato*, ed. Hamilton and Cairns, 149d, 28, 24-25.
- Cook, S. D. N., & Brown, J. S. (1999). Bridging epistemologies: The generative dance between organizational knowledge and organizational knowing. *Organization Science*, 10(4), 381-400. <https://doi.org/10.1287/orsc.10.4.381>
- Copello, L., Porteron, S., & Schweitzer, J.-P. (2021). Realising reuse - The potential for scaling up reusable packaging, and policy recommendations. In R. P. Alliance (Ed.). Brussels: Rethink Plastic Alliance.

- Corrado, C., Haskel, J., Miranda, J., & Sichel, D. (2021). *Measuring and accounting for innovation in the twenty-first century* (Vol. 78). University of Chicago Press. <https://doi.org/10.7208/chicago/9780226728209.001.0001>
- Corrado, C. A., & Hulten, C. R. (2013). *Internationalization of intangibles* Globalization and U.S. Statistics, Washington, D.C. <https://www.upjohn.org/MEG/papers/Corrado-Hulten.pdf>
- Creswell, J. W. (2007). An introduction to mixed methods research. In *SSP, University of Nebraska-Lincoln* (pp. 43). University of Nebraska-Lincoln.
- Custers, B., & Uršič, H. (2016). Big data and data reuse: a taxonomy of data reuse for balancing big data benefits and personal data protection. *International Data Privacy Law*, 6(1), 4-15. <https://doi.org/10.1093/idpl/ipv028>
- Cusumano, M. A. (1991). Systematic' versus 'accidental' reuse in Japanese software factories. *Sloan School of Management*(WP# 3328-BPS-91), 1-26.
- Davenport, T. H., & Prusak, L. (2000). Working knowledge - how organizations manage what they know. *Ubiquity*, 2000(August), 2-16. <https://doi.org/10.1145/347634.348775>
- Davis, M. J. (1992). STARS reuse maturity model: Guidelines for reuse strategy formulation. Proceedings of the Fifth Annual Workshop on Institutionalizing Software Reuse, Palo Alto, CA.
- de Holan, P. M., & Phillips, N. (2004). Remembrance of Things Past? The Dynamics of Organizational Forgetting [research-article]. *Management Science*, 50(11), 1603 - 1613. <https://doi.org/10.1287/mnsc.1040.0273>
- Deakin, H., & Wakefield, K. (2014). Skype interviewing: reflections of two PhD researchers. *Qualitative Research*, 14(5), 603-616. <https://doi.org/10.1177/1468794113488126>
- Definitions.net. (2022). *The Web's Largest Resource for Definitions & Translations*. The STANDS4 Network. Retrieved 2022-09-07 from <https://www.definitions.net/definition/ASSET>
- Del Vecchio, L. (2022). The Sharing Economy: shifting from ownership to a sharing paradigm. *techDetector*. <https://techdetector.de/stories/thesharingeconomy#from-linearity-to-circularity>
- DeLone, W. H., & McLean, E. R. (2003). The DeLone and McLean model of information systems success: a ten-year update. *Journal of management information systems*, 19(4), 9-30. <https://doi.org/10.1080/07421222.2003.11045748>
- Devanbu, P., Brachman, R., Selfridge, P. G., & Ballard, B. W. (1991). LaSSIE. *Communications of the ACM*, 34(5), 34-49. <https://doi.org/10.1145/103167.103172>
- Di Cosmo, R., & Zacchiroli, S. (2017). *Software heritage: Why and how to preserve software source code* iPRES 2017 - 14th international conference on digital preservation, Kyoto, Japan.

- Dick, R. (1993). *You Want to Do an Action Research Thesis?: How to Conduct and Report Action Research, (including a Beginner's Guide to the Literature)*. Interchange.
- Dixit, A. K., & Pindyck, R. S. (1995). The options approach to capital investment. *Real Options and Investment under Uncertainty-classical Readings and Recent Contributions*, 6(May-June 1995).
- Dong, X. L., Gabrilovich, E., Murphy, K., Dang, V., Horn, W., Lugaresi, C., . . . Zhang, W. (2015). Knowledge-based trust. *Proceedings of the VLDB Endowment*, 8(9), 938-949. <https://doi.org/10.14778/2777598.2777603>
- Dreyer, H., & Wynn, M. G. (2016). Tacit and explicit knowledge in software development projects: A combined model for analysis. *International Journal on Advances in Software*, 9(3/4), 154-166.
- Drucker, P. F. (1957). *The landmarks of tomorrow*. Harper & Row.
- Eady, S., Drew, V., & Smith, A. (2015). Doing action research in organizations: Using communicative spaces to facilitate (transformative) professional learning. *Action Research*, 13(2), 105-122. <https://doi.org/10.1177/1476750314549078>
- Eid, M., & Nuhu, N. A. (2011). Impact of learning culture and information technology use on knowledge sharing of Saudi students. *Knowledge Management Research & Practice*, 9(1), 48-57. <https://doi.org/10.1057/kmrp.2010.25>
- Eilbert, K. W., & Lafronza, V. (2005). Working together for community health—a model and case studies. *Evaluation and Program Planning*, 28(2), 185-199. <https://doi.org/10.1016/j.evalprogplan.2005.01.003>
- Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *Academy of Management Review*, 14(4), 532-550. <https://doi.org/10.5465/amr.1989.4308385>
- Eisenhardt, K. M. (2020). Theorizing from Cases: A Commentary. In L. Eden, B. Nielsen, & A. Verbeke (Eds.), *Research methods in international business* (pp. 221-227). Palgrave Macmillan. https://doi.org/10.1007/978-3-030-22113-3_10
- El Baff, R., Santhanam, S., & Hecking, T. (2021). Quantifying synergy between software projects using README files only. Proceedings of the 33rd International Conference on Software Engineering and Knowledge Engineering,
- Enders, A., Koenig, A., & Barsoux, J.-L. (2016). Stop jumping to solutions! *MIT Sloan Management Review*, 57(4), 63-70.
- England, P., & Folbre, N. (2023). Reconceptualizing human capital. In *A Research Agenda for Skills and Inequality* (pp. 177-195). Edward Elgar Publishing. <https://doi.org/10.4337/9781800378469.00017>
- Epley, N., Waytz, A., & Cacioppo, J. T. (2007). On seeing human: A three-factor theory of anthropomorphism. *Psychological review*, 114(4), 864-886. <https://doi.org/10.1037/0033-295X.114.4.864>

- Fadler, M., & Legner, C. (2020, 09.10.2019). *Managing data as an asset with the help of artificial intelligence* SAP Business Data and Analytics Info Day event, Essen (Germany). <https://www.cc-cdq.ch/news/whitepaper-managing-data-as-an-asset>
- Fagerberg, J. (2003). Schumpeter and the revival of evolutionary economics: an appraisal of the literature. *Journal of Evolutionary Economics*, 13(2), 125-159. <https://doi.org/10.1007/s00191-003-0144-1>
- Fakhar Manesh, M., Pellegrini, M. M., Marzi, G., & Dabic, M. (2021). Knowledge management in the fourth industrial revolution: Mapping the literature and scoping future avenues. *IEEE Transactions on Engineering Management*, 68(1), 289-300. <https://doi.org/10.1109/tem.2019.2963489>
- Fanchao, M., Dechen, Z., & Xiaofei, X. (2006). A specification-based approach for retrieval of reusable business component for software reuse. *International Journal of Computer Science*, 1(1), 240-247. <https://doi.org/10.5281/zenodo.1075773>
- Fayyaz, A., Chaudhry, B. N., & Fiaz, M. (2020). Upholding knowledge sharing for organization innovation efficiency in Pakistan [Article]. *Journal of Open Innovation: Technology, Market, and Complexity*, 7(4), 1-17. <https://doi.org/10.3390/joitmc7010004>
- Fichman, R. G., & Kemerer, C. F. (2001). Incentive compatibility and systematic software reuse. *The Journal of systems and software*, 57(2001), 45-60. [https://doi.org/10.1016/S0164-1212\(00\)00116-3](https://doi.org/10.1016/S0164-1212(00)00116-3)
- Fivet, C., & Brütting, J. (2020). Nothing is lost, nothing is created, everything is reused: structural design for a circular economy. *The Structural Engineer*, 98(1), 74-81.
- Flath, C. M., Friesike, S., Wirth, M., & Thiesse, F. (2017). Copy, transform, combine: Exploring the remix as a form of innovation. *Journal of Information Technology*, 32(4), 306-325. <https://doi.org/10.1057/s41265-017-0043-9>
- Flores, E., Xu, X., & Lu, Y. (2020). Human capital 4.0: A workforce competence typology for Industry 4.0. *Journal of manufacturing technology management.*, 31(4), 687-703. <https://doi.org/10.1108/JMTM-08-2019-0309>
- Flyvbjerg, B. (2011). Five misunderstandings about case-study research. *Qualitative Inquiry*, 12(2), 219-245. <https://doi.org/10.1177/1077800405284363>
- Fogg, B. (2009). A behavior model for persuasive design. *Proceedings of the 4th International Conference on Persuasive Technology - Persuasive '09*, 1-7. <https://doi.org/10.1145/1541948.1541999>
- Fortune, J., & Valerdi, R. (2013). A framework for reusing systems engineering products. *Systems Engineering*, 16(3), 304-312. <https://doi.org/10.1002/sys.21232>
- Frakes, W., & Terry, C. (1996). Software reuse: Metrics and models. *ACM Computing Surveys*, 28(2), 415-435. <https://doi.org/10.1145/234528.234531>

- Frakes, W. B., Biggerstaff, T. J., Prieto-Diaz, R., Matsumura, K., & Schaefer, W. (1991). Software reuse: is it delivering? 13th International Conference on Software Engineering,
- Frakes, W. B., & Fox, C. J. (1995). Sixteen questions about software reuse. *Communications of the ACM*, 38(6), 75-87.
<https://doi.org/10.1145/203241.203260>
- Frakes, W. B., & Nejme, B. A. (1986). Software reuse through information retrieval. *ACM SIGIR Forum*, 21(1-2), 30-36.
<https://doi.org/10.1145/24634.24636>
- Fraser, S. (2021). Five strategies for the future of work: Accelerating innovation through tech transfer. *Agile Alliance*, 1-8.
- Fu, X., & Ghauri, P. (2021). Trade in intangibles and the global trade imbalance. *The World Economy*, 44(5), 1448-1469.
<https://doi.org/10.1111/twec.13038>
- Gagne, M., Tian, A., Soo, C., Zhang, B., Ho, K. S. B., & Hosszu, K. (2019). Why employees don't share knowledge with each other. *Harvard Business Review*(Leading Teams), 1-9.
- Gall, H. (2004). *Kapitel 12 Software Wiederverwendung*. Universität Zürich, Institut für Informatik. Retrieved 2021-12-27 from
https://files.ifi.uzh.ch/rerg/amadeus/teaching/courses/kvse_ss05/kap12-reuse.pdf
- Gamble, J. R. (2020). Tacit vs explicit knowledge as antecedents for organizational change. *Journal of organizational change management.*, 33(6), 1123-1141.
<https://doi.org/10.1108/JOCM-04-2020-0121>
- Garcia, V. C., Lisboa, L. B., Almeida, E. S. D., Meira, S. R. D. L., Almeida, E. S. D., Lucrédio, D., & Fortes, R. P. D. M. (2008, 2008-08-01). Towards an Assessment Method for Software Reuse Capability (Short Paper). 2008 The Eighth International Conference on Quality Software, Washington, DC.
- Garcia, V. C., Lucrédio, D., Alvaro, A., De Almeida, E. S., de Mattos Fortes, R. P., & de Lemos Meira, S. R. (2007). Towards a maturity model for a reuse incremental adoption. SBCARS, Campinas, Brazil.
- Geissdoerfer, M., Pieroni, M.P., Pigosso, D.C. and Soufani, K. (2020). Circular business models: A review. *Journal of Cleaner Production*, 277(123741), 1 - 17. <https://doi.org/10.1016/j.jclepro.2020.123741>
- Geology.com. (2021). *World Map: A clickable map of world countries*. Geology.com. Retrieved 2021-12-29 from <https://geology.com/world/world-map.shtml>
- Goffman, E. (1981). *Forms of talk*. University of Pennsylvania Press.
- Goldkuhl, G. (2012). Pragmatism vs interpretivism in qualitative information systems research. *European Journal of Information Systems*, 21(2), 135-146. <https://doi.org/10.1057/ejis.2011.54>
- Goodridge, P., Haskel, J., & Wallis, G. (2016). *UK intangible investment and growth: New measures of UK investment in knowledge assets and intellectual property rights*. Intellectual Property Office.

- Gouinlock, J. (1995). The moral writings of John Dewey, Revised edition [The Moral Writings of John Dewey, Revised Edition, James Gouinlock]. *Metaphilosophy*, 26(4), 431-435. <https://doi.org/10.1111/j.1467-9973.1995.tb00587.x>
- Gourlay, S. (2006). Conceptualizing knowledge creation: A critique of Nonaka's theory. *Journal of Management Studies*, 43(7), 1415-1436. <https://doi.org/10.1111/j.1467-6486.2006.00637.x>
- Griss, M. (1996, October 1996). *Systematic software reuse: Architecture, process and organization are crucial*. Retrieved 2022-09-10 from <http://martin.griss.com/pubs/fusion1.htm>
- Griss, M. L. (1993). Software reuse: From library to factory. *IBM Systems Journal*, 32(4), 548-566. <https://doi.org/10.1147/sj.324.0548>
- Griss, M. L. (2015). Systematic software reuse - it isn't what it used to be. ISCR, Pittsburgh, PA.
- Griss, M. L., & Wosser, M. (1995). Making reuse work at Hewlett-Packard. *IEEE Software*, 12(1), 105-107. <https://doi.org/10.1109/52.363160>
- Génova, G., Llorens, J., Metz, P., Prieto-Díaz, R., & Astudillo, H. (2005). Open issues in industrial use case modeling. In N. J. Nunes (Ed.), *UML Modeling Languages and Applications* (pp. 52-61). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-31797-5_6
- Hargadon, A. B. (2002). Brokering knowledge: Linking learning and innovation. *Research in Organizational behavior*, 24, 41-85. [https://doi.org/10.1016/S0191-3085\(02\)24003-4](https://doi.org/10.1016/S0191-3085(02)24003-4)
- Haskel, J., & Westlake, S. (2018). *Capitalism without capital: The rise of the intangible economy*. Princeton University Press. <https://doi.org/10.2307/j.ctvc77hhj>
- Haskel, J., & Westlake, S. (2022). Restarting the Future. In *Restarting the Future* (pp. 320). Princeton University Press. <https://doi.org/10.1515/9780691236025>
- Hasnain, H., & Mohseni, F. (2018). Creative ideation and adaptive reuse: a solution to sustainable urban heritage conservation. *Earth Environment IOP conference series: earth and environmental science*, Medan, Indonesia.
- Hasselbring, W., Carr, L., Hettrick, S., Packer, H., & Tiropanis, T. (2020). From FAIR research data toward FAIR and open research software. *it - Information Technology*, 62(1), 39-47. <https://doi.org/10.1515/itit-2019-0040>
- Hedlund, G. (1994). A model of knowledge management and the N-form corporation. *Strategic Management Journal*, 15(S2), 73-90. <https://doi.org/10.1002/smj.4250151006>
- Hejase, H. J., Haddad, Z., Hamdar, B., Al Ali, R., Hejase, A. J., & Beyrouti, N. (2014). Knowledge sharing: Assessment of factors affecting employee' motivation and behavior in the Lebanese organizations. *Journal of Scientific Research and Reports*, 3(12), 1549-1593. <https://doi.org/10.9734/JSRR/2014/8107>

- Henry, G. T., Smith, A. A., Kershaw, D. C., & Zulli, R. A. (2013). Formative evaluation: Estimating preliminary outcomes and testing rival explanations. *American Journal of Evaluation, 34*(4), 465-485. <https://doi.org/0.1177/1098214013502577>
- Herriott, R. E., & Firestone, W. A. (1983). Multisite qualitative policy research: Optimizing description and generalizability. *Educational researcher, 12*(2), 14-19. <https://doi.org/10.3102/0013189X012002014>
- Hjørland, B. (2013). Theories of knowledge organization — theories of knowledge. *Knowledge Organization, 40*(3), 169-181. <https://doi.org/10.5771/0943-7444-2013-3-169>
- Hofstetter, W. K. (2009). *A framework for the architecting of aerospace systems portfolios with commonality* [Massachusetts Institute of Technology]. Cambridge, MA. <http://hdl.handle.net/1721.1/50598>
- Holibaugh, R., Cohen, S., Kang, K., & Peterson, S. (1989). Reuse: where to begin and why. Proceedings of the conference on Tri-Ada'89: Ada technology in context: application, development, and deployment, *Pittsburgh, PA*.
- Horbach, J., & Rammer, C. (2020). Circular economy innovations, growth and employment at the firm level: Empirical evidence from Germany. *Journal of Industrial Ecology, 24*(3), 615-625. <https://doi.org/10.1111/jiec.12977>
- Horne, N. W. (1995). Information as an asset—The board agenda. *Computer Audit Update, 1995*(9), 5-11.
- Hsieh, H.-F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative health research, 15*(9), 1277-1288. <https://doi.org/10.1177/1049732305276687>
- Huang, S. (2019, 2019-02-27). The other side of technical skill: Domain knowledge and long-term vision. *Sihui Huang - Career Development*. <https://www.sihui.io/domain-knowledge-and-vision/>
- Huber Jr, B. J. (1991). *A Knowledge-based approach to understanding natural language* [Rochester Institute of Technology]. Henrietta, NY.
- Human, G. (2020). Linking absorptive capacity, knowledge transfer and transactive memory. *Journal of business & industrial marketing., 26*(10), 1740-1754. <https://doi.org/10.1108/JBIM-01-2020-0060>
- Hussinki, H., Ritala, P., Vanhala, M., & Kianto, A. (2017). Intellectual capital, knowledge management practices and firm performance. *Journal of Intellectual Capital, 18*(4), 904-922. <https://doi.org/10.1108/jic-11-2016-0116>
- IFRS. (2022). *IAS 38 Intangible assets*. IFRS. Retrieved 2022-09-13 from <https://www.ifrs.org/issued-standards/list-of-standards/ias-38-intangible-assets/>
- Imoize, A. L., Idowu, D., & Bolaji, T. (2019). A brief overview of software reuse and metrics in software engineering. *World Scientific News, 122*, 56-70.
- In, J. (2017). Introduction of a pilot study. *Korean Journal of Anesthesiology, 70*(6), 601. <https://doi.org/10.4097/kjae.2017.70.6.601>

- Irick, M. L. (2007). Managing tacit knowledge in organizations. *Journal of Knowledge Management Practice*, 8(3), 1-5.
<https://doi.org/10.1109/ICDIM.2008.4746707>
- Irshad, M. (2010). *Measuring cost avoidance through software reuse* (Publication Number MSE-2010-38) Blekinge Institute of Technology]. Karlskrona, Sweden.
- Irshad, M., Petersen, K., & Poulding, S. (2018). A systematic literature review of software requirements reuse approaches. *Information and software technology*, 93, 223-245. <https://doi.org/10.1016/j.infsof.2017.09.009>
- Irshad, M., Torkar, R., Petersen, K., & Afzal, W. (2016, 2016-06-01). Capturing cost avoidance through reuse. Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, Limerick, Ireland.
- Ismail, K. (2019, 2022-07-09). SaaS vs. Cloud: Comparing Apples and Oranges. *CMSWire*. <https://www.cmswire.com/information-management/saas-vs-cloud-comparing-apples-and-oranges/>
- Jalender, B., Govardhan, A., & Premchand, P. (2010). A pragmatic approach to software reuse. *Journal of Theoretical & Applied Information Technology*, 14.
- James, W. (1907). Lecture II: What pragmatism means. In W. James (Ed.), *Pragmatism: A new name for some old ways of thinking* (pp. 43-81). Longmans, Green and Co. <https://doi.org/10.1037/10851-002>
- Jarikre, A. O., Sharma, Y. K., John, A. K., & Bailey, S. K. (2022). Analysing the obstacles in agile software development approach: A review. *East African Journal of Information Technology*, 5(1), 1-6.
<https://doi.org/10.37284/eajit.5.1.520>
- Jarvis, J. (2009). *What would Google do?* (Vol. 48). HarperCollins Publishers Limited.
- Jasimuddin, S. M. (2005). An integration of knowledge transfer and knowledge storage: An holistic approach. *Comput Sci Eng*, 18(1), 37-49.
- Jasin, M., Anisah, H. U., Fatimah, C. E. A., Azra, F. E. A., Suzanawaty, L., & Junaedi, I. W. R. (2024). The role of digital literacy and knowledge management on process innovation in SMEs. *International journal of data and network science*, 8(1), 337-344.
<https://doi.org/10.5267/j.ijdns.2023.9.020>
- Johnson, E. S. (2020). *Action research*. Oxford University Press.
<https://doi.org/10.1093/acrefore/9780190264093.013.696>
- Jones, C. I., & Tonetti, C. (2020). Nonrivalry and the economics of data. *American Economic Review*, 110(9), 2819-2858.
<https://doi.org/10.1257/aer.20191330>
- Jones, G., & Prieto-Diaz, R. (1988). Building and managing software libraries. Proceedings COMPSAC 88: The twelfth annual international computer software & applications conference, Chicago, IL, USA.

- Jonsson, C.-J., Stolt, R., & Elgh, F. (2021). Design Knowledge Reuse in Design of Progressive Stamping Tools: A Qualitative Study. *Proceedings of the Design Society, 1*, 1053-1062. <https://doi.org/10.1017/pds.2021.105>
- Kahn, K. B. (2018). Understanding innovation. *Business Horizons, 61*(3), 453-460. <https://doi.org/10.1016/j.bushor.2018.01.011>
- Kallio, H., Pietilä, A.-M., Johnson, M., & Kangasniemi, M. (2016). Systematic methodological review: Developing a framework for a qualitative semi-structured interview guide. *Journal of Advanced Nursing, 72*(12), 2954-2965. <https://doi.org/10.1111/jan.13031>
- Karim, D., & Majid, A. (2018). Autonomous job design and knowledge sharing behavior: The mediating role of public service motivation. *VI*(4), 607-622.
- Karimpanal, T. G., & Bouffanais, R. (2019). Self-organizing maps for storage and transfer of knowledge in reinforcement learning. *Adaptive Behavior, 27*(2), 111-126. <https://doi.org/10.1177/1059712318818568>
- Kaushik, V., & Walsh, C. A. (2019). Pragmatism as a research paradigm and its implications for social work research. *Social Sciences, 8*(9), 255. <https://doi.org/10.3390/socsci8090255>
- Kelleher, M. (2018, 2018-08-13). Let me tell you a story: How to facilitate the sharing of tacit knowledge. <https://www.linkedin.com/pulse/let-me-tell-you-story-how-facilitate-sharing-tacit-michael-kelleher/>
- Kessel, M., & Atkinson, C. (2015, 2015-05-01). Ranking software components for pragmatic reuse. 2015 IEEE/ACM 6th International Workshop on Emerging Trends in Software Metrics, Florence, Italy.
- Keswani, R., Joshi, S., & Jatain, A. (2014, 2014-02-01). Software reuse in practice. 2014 Fourth International Conference on Advanced Computing & Communication Technologies, Rohtak, India.
- Khavandkar, E., Theodorakopoulos, N., Hart, M., & Preston, J. (2016). Leading the diffusion of intellectual capital management practices in science parks. In H. Shipton, P. Budhwar, P. Sparrow, & A. Brown (Eds.), *Human resource management, innovation and performance* (pp. 213-231). Palgrave Macmillan UK. https://doi.org/10.1057/9781137465191_14
- Khisty, C. J. (2000). Can wicked problems be tackled through abductive inferencing? *Journal of urban planning and development, 126*(3), 104-118. [https://doi.org/10.1061/\(ASCE\)0733-9488\(2000\)126:3\(104\)](https://doi.org/10.1061/(ASCE)0733-9488(2000)126:3(104))
- Kiran, K. T. R. B. R., & Vindhya, A. (2011). A systematic mapping study on value of reuse. *International Journal of Computer Applications, 34*(4), 37-44.
- Kneuper, R. (2002). Supporting software processes using knowledge management. In S.-K. Chang (Ed.), *Handbook of software engineering and knowledge engineering: Volume II: Emerging Technologies* (pp. 579-606). World Scientific Publishing Company.
- Koeh Sitienei, C., Boit, J. M., & Maru, L. (2015). Knowledge storage, retrieval, and employee performance: The moderating role of employee engagement. *International Journal of Small Business and Entrepreneurship Research, 3*(6), 1-13.

- Koloniari, M., Vraimaki, E., & Fassoulis, K. (2019). Factors affecting knowledge creation in academic libraries. *Journal of Librarianship and Information Science*, 51(1), 20-33. <https://doi.org/10.1177/0961000616668958>
- Konstantara, M. K., & Galanakis, M. (2022). Organizational & Industrial Psychology in the 21st Century—Goal-Setting Theory and Performance Management: A Systematic Literature Review. *Psychology*, 13(05), 790-797. <https://doi.org/10.4236/psych.2022.135052>
- KPMG. (2018). Let's help SMEs to go circular. In KPMG (Ed.), *Project: Boosting the circular economy amongst SMEs in Europe* (Training Modules ed., pp. 136). Netherlands: A project of the European Commission - DG Environment.
- Krishnaveni, R., & Raja, C. S. (2012). Factors that enable knowledge management in information technology organizations – a statistical study. *Journal of Contemporary research in Management*, 3(4), 81-100.
- Krueger, C. W. (1992). Software reuse. *ACM Computing Surveys*, 24(2), 131-183. <https://doi.org/10.1145/130844.130856>
- Krüger, J., & Berger, T. (2020, 2020-11-08). An empirical analysis of the costs of clone- and platform-oriented software reuse. Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual event, USA.
- Kuo, T. H. (2013). How expected benefit and trust influence knowledge sharing. *Industrial Management & Data Systems*, 113(4), 506-522. <https://doi.org/10.1108/02635571311322766>
- Lacy, P., Long, J., & Spindler, W. (2020). *The Circular Economy Handbook: Realizing the Circular Advantage*. Palgrave Macmillan. <https://doi.org/10.1057/978-1-349-95968-6>
- Lai, C. (1981). The role of practice in the Marxist theory of knowledge. *The 80's*, II(1), 1-28.
- Lam, A. (1997). Embedded firms, embedded knowledge: Problems of collaboration and knowledge transfer in global cooperative ventures. *Organization studies*, 18(6), 973-996.
- Lange, C. (2020). *An analysis of design reuse in the development of small body nanolander and surface science stations: Supporting reuse, modular design and platforming with Model Based Systems Engineering* [Universität Bremen]. Bremen, Germany.
- Langley, P. (2020). Assets and assetization in financialized capitalism. *Review of International Political Economy*, 28(2), 382-393. <https://doi.org/10.1080/09692290.2020.1830828>
- Lear, J. (2021). *The Video Game Asset Pipeline A Pattern Approach to Visualization* [The University of the West of England]. Bristol, UK.
- Leber, M., Buchmeister, B., & Ivanisevic, A. (2015). Impact of knowledge on innovation process. In B. Katalinic (Ed.), *DAAAM International Scientific Book 2015* (pp. 235-248). DAAAM International Vienna. <https://doi.org/10.2507/daaam.scibook.2015.21>

- Lev, B. (2018). The deteriorating usefulness of financial report information and how to reverse it [research-article]. *Accounting and Business Research*, 48(5: International Accounting Policy Forum), 465-493. <https://doi.org/10.1080/00014788.2018.1470138>
- Lindgreen, A., Di Benedetto, C. A., & Beverland, M. B. (2021). How to write up case-study methodology sections. *Industrial Marketing Management*, 96(July 2021), A7-A10. <https://doi.org/10.1016/j.indmarman.2020.04.012>
- Liu, H., Li, L., Lu, S., Zhang, K., & Liu, X. (2020). 3D model similarity evaluation for mechanical design reuse based on spatial correlated shape-word clique. *Multimedia Tools and Applications*, 79, 8181-8195. <https://doi.org/10.1007/s11042-019-08315-4>
- Lloyd, R., & Mertens, D. (2018). Expecting more out of expectancy theory: History urges inclusion of the social context. *International Management Review*, 14(1), 28-43.
- Lobe, B., Morgan, D., & Hoffman, K. A. (2020). Qualitative Data Collection in an Era of Social Distancing. *International Journal of Qualitative Methods*, 19, 1-8. <https://doi.org/10.1177/1609406920937875>
- Maccanti, S., Al-Jaroodi, J., & Sirinterlikci, A. (2016). Knowledge management framework for software reuse. 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), Atlanta, GA, USA.
- MacDonald, S., & Headlam, N. (2008). *Research methods handbook: Introductory guide to research methods for social research*. Centre for Local Economic Strategies.
- MacLeod, H., & Sommerville, D. (2016, 2016-05-22). *Awesome graphic as graphs*. Retrieved 2021-08-01 from <http://kerfors.blogspot.com/2016/05/awesome-graphic-as-graphs.html>
- Majchrzak, A., Cooper, L. P., & Neece, O. E. (2004). Knowledge reuse for innovation. *Management science*, 50(2), 174-188. <https://doi.org/10.1287/mnsc.1030.0116>
- Majchrzak, A., Neece, O. E., & Cooper, L. P. (2001). Knowledge reuse for innovation - The missing focus in knowledge management: Results of a case analysis at the jet propulsion laboratory. *Academy of Management Proceedings*, Briarcliff Manor, NY, USA.
- Manjhi, D., & Chaturvedi, A. (2019, 2019-02-01). Software component reusability classification in functional paradigm. 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India.
- Markus, M. L. (2001). Toward a theory of knowledge reuse: Types of knowledge reuse situations and factors in reuse success. *Journal of Management Information Systems*, 18(1), 57-93. <https://doi.org/10.1080/07421222.2001.11045671>
- Marrucci, L., Daddi, T., & Iraldo, F. (2022). Do dynamic capabilities matter? A study on environmental performance and the circular economy in European certified organisations. *Business strategy and the environment*, 31(6), 2641-2657. <https://doi.org/10.1002/bse.2997>

info:doi/10.1002/bse.2997

- Masa'deh, R. e., Almajali, D. A., Alrowwad, A. a., & Obeidat, B. (2019). The role of knowledge management infrastructure in enhancing job satisfaction: A developing country perspective. *Interdisciplinary Journal of Information, Knowledge & Management*, 14, 1-25. <https://doi.org/10.28945/4169>
- Matallo, J. H. (2021). The value of case studies and traditional knowledge for a sustainable future. *Academia Letters*, 1-5, Article Article 1880. <https://doi.org/https://doi.org/10.20935/AL1880>
- Mateen, A., Kausar, S., & Sattar, A. R. (2017). A software reuse approach and its effect on software quality, an empirical study for the software industry. *International Journal of Management, IT & Engineering*, 7(2), 266-279. <https://doi.org/10.48550/arXiv.1702.00125>
- Mauboussin, M. J., & Callahan, D. (2020). Expectations and the role of intangible investments. *Morgan Stanley Investment Management*(Consilient Observer), 1-25. Retrieved 2022-09-17, from https://www.morganstanley.com/im/publication/insights/articles/articles_on_ejob.pdf <https://acquirersmultiple.com/2020/09/michael-mauboussin-one-job-expectations-and-the-role-of-intangible-investments/>
- McGrath, C., Palmgren, P. J., & Liljedahl, M. (2019). Twelve tips for conducting qualitative research interviews. *Medical Teacher*, 41(9), 1002-1006. <https://doi.org/10.1080/0142159x.2018.1497149>
- McGuire, K. L. (2022). Methods of exploring related-meaning-based false memories. *Journal of Cognition and Development*, 23(1), 64-82. <https://doi.org/10.1080/15248372.2021.1976782>
- McIlroy, M. D., Buxton, J., Naur, P., & Randell, B. (1968). Mass-produced software components. Proceedings of the 1st international conference on software engineering, Garmisch Pattenkirchen, Germany.
- McIntosh, M. J., & Morse, J. M. (2015). Situating and constructing diversity in semi-structured interviews. *Global Qualitative Nursing Research*, 2(0), 1-12. <https://doi.org/10.1177/2333393615597674>
- McMahon, C., Lowe, A., & Culley, S. (2004). Knowledge management in engineering design: Personalization and codification. *Journal of Engineering Design*, 15(4), 307-325. <https://doi.org/10.1080/09544820410001697154>
- Mehta, A. D., & Madhani, P. M. (2008). Intangible assets - an introduction. *The Accounting World*, 8(9), 11-19.
- Merriam-Webster. (2021a). Definition of REUSE. In *Merriam-Webster, since 1828*. Retrieved 2021-10-09, from <https://www.merriam-webster.com/dictionary/reuse>
- Merriam-Webster. (2021b). systematic. In *Merriam-Webster, since 1828*. Retrieved 2021-05-03, from <https://www.merriam-webster.com/dictionary/systematic>
- Mignacca, B., Locatelli, G., & Velenturf, A. (2020). Modularisation as enabler of circular economy in energy infrastructure. *Energy Policy*, 139(111371), 1-11. <https://doi.org/10.1016/j.enpol.2020.111371>

- Mihale-Wilson, C., Felka, P., Hinz, O., & Spann, M. (2021). The impact of strategic core-component reuse on product life cycles. *Business & Information Systems Engineering*, 64(2), 223-237. <https://doi.org/10.1007/s12599-021-00706-y>
- Milala, S. I., Ariffin, K. M., Kasim, R., Yassin, A. M., Ishak, M. H., & Kasim, N. (2021). Intangible asset a key driver for company's performance: An overview. International Conference on Industrial Engineering and Operations Management, Rome, Italy.
- Mohagheghi, P., & Conradi, R. (2007). Quality, productivity and economic benefits of software reuse: A review of industrial studies. *Empirical Software Engineering*, 12(5), 471-516. <https://doi.org/10.1007/s10664-007-9040-x>
- Montani, G. (1987). Scarcity. In J. Eatwell, M. Milgate, & P. Newman (Eds.), *The new Palgrave: A dictionary of economics* (pp. 1-4). Palgrave Macmillan. https://doi.org/10.1057/978-1-349-95121-5_1318-1
- Mulkay, M. (2014). *Science and the sociology of knowledge* (Vol. 60). Routledge. <https://doi.org/10.4324/9781315763408>
- Mäkitalo, N., Taivalsaari, A., Kiviluoto, A., Mikkonen, T., & Capilla, R. (2020). On opportunistic software reuse. *Computing*, 102(11), 2385-2408. <https://doi.org/10.1007/s00607-020-00833-6>
- Nakamori, Y. (2020). Knowledge management. In *Translational systems sciences* (pp. 63-91). Springer. https://doi.org/10.1007/978-981-13-9887-2_4
- Nakamori, Y. (2021). Converting information into knowledge. In *Knowledge technology* (pp. 57-73). SpringerBriefs in Business. https://doi.org/10.1007/978-981-16-3253-2_5
- Newsroom, I. (2009, 2009-12-03). *IBM and Vanderlande industries sign agreement for baggage system at Amsterdam Schiphol airport*. IBM. Retrieved 2021-12-12 from <https://newsroom.ibm.com/2009-12-03-IBM-and-Vanderlande-Industries-Sign-Agreement-for-Baggage-System-at-Amsterdam-Schiphol-Airport>
- Newsroom, I. (2017, REPLACE). *Deutsche Post DHL Group und IBM: Pakete sortieren in Rekordzeit*. IBM. Retrieved 2021-12-12 from <https://de.newsroom.ibm.com/2017-01-30-Deutsche-Post-DHL-Group-und-IBM-Pakete-sortieren-in-Rekordzeit>
- Nickols, F. (2000). The knowledge in knowledge management. *The knowledge management yearbook, 2000-2001*, 12, 1-8.
- Noble, H., & Smith, J. (2015). Issues of validity and reliability in qualitative research. *Evidence-based nursing*, 18(2), 34-35. <https://doi.org/10.1136/eb-2015-102054>
- Nonaka, I. (1991). The knowledge-creating company. *Harvard business review*, 85(7/8), 162-171.
- Nonaka, I. (1994). A dynamic theory of organizational knowledge creation. *Organization Science*, 5(1), 3-42. <https://doi.org/10.1016/b978-0-7506-7111-8.50003-2>

- Nonaka, I., & Teece, D. (2001). *Managing industrial knowledge: Creation, transfer and utilization* (Vol. 3). SAGE publications. <https://doi.org/10.1002/kpm.183>
- Nowell, L. (2015). Pragmatism and integrated knowledge translation: Exploring the compatibilities and tensions. *Nursing Open*, 2(3), 141-148. <https://doi.org/10.1002/nop2.30>
- Nozick, R. (1981). *Philosophical explanations*. Harvard University Press.
- Obeidat, U., Obeidat, B., Alrowwad, A., Alshurideh, M., Masadeh, R., & Abuhashesh, M. (2021). The effect of intellectual capital on competitive advantage: The mediating role of innovation. *Management Science Letters*, 11(4), 1331-1344. <https://doi.org/10.5267/j.msl.2020.11.006>
- Okosun, J. (2011). Analysing the cause and effect of IS failures in complex projects: Case Study of Heathrow Terminal 5. *ResearchGate*(December), 1-14.
- Oppenheim, C. (1998). Valuing information assets in British companies. *Business Information Review*, 15(4), 209-214. <https://doi.org/10.1177/0266382984236920>
- Oppenheim, C., Stenson, J., & Wilson, R. M. S. (2003). Studies on Information as an Asset I: Definitions. *Journal of Information Science*, 29(3), 159-166. <https://doi.org/10.1177/01655515030293003>
- Ormerod, R. J. (2020). Pragmatism in professional practice. *Systems Research and Behavioral Science*, 38(6), 797-816. <https://doi.org/10.1002/SRES.2739>
- Ostertag, E. J., Hendler, J. A., Prieto-Diaz, R., & Braun, C. (1991). *Computer similarity in a reuse library system: An AI-based approach*.
- Paoloni, M., Coluccia, D., Fontana, S., & Solimene, S. (2020). Knowledge management, intellectual capital and entrepreneurship: A structured literature review. *Journal of knowledge management*, 24(8), 1797-1818. <https://doi.org/10.1108/JKM-01-2020-0052>
- Parikh, R., & Renero, A. (2017). Justified true belief: Plato, Gettier, and Turing. In J. Floyd & A. Bokulich (Eds.), *Boston Studies in the Philosophy and History of Science* (pp. 93-102). Springer International Publishing. https://doi.org/10.1007/978-3-319-53280-6_4
- Park, A., Maine, E., Fini, R., Rasmussen, E., Di Minin, A., Dooley, L., . . . Zhou, Y. (2023). Science-based Innovation via University Spin-offs: A Systematic Review of Intangible Assets. *R&D Management*, 1-21. <https://doi.org/http://dx.doi.org/10.1111/radm.12646>
- Pasquetto, I. V., Borgman, C. L., & Wofford, M. F. (2019). Uses and reuses of scientific data: The data creators' advantage. *Harvard Data Science Review*, 1(2), 1-32. <https://doi.org/10.1162/99608f92.fc14bf2d>
- Percival, R. S. (1996). The metaphysics of scarcity: Popper's world 3 and the theory of finite resources. *The Critical Rationalist*, 1(2), 32.
- Pitt, C. A. (2007). *Leading innovation and entrepreneurship: An action research study in the Australian red meat industry* Southern Cross University]. Lismore, Australia.

- Polit, D. F., & Beck, C. T. (2010). Generalization in quantitative and qualitative research: Myths and strategies. *International Journal of Nursing Studies*, 47(11), 1451-1458. <https://doi.org/10.1016/j.ijnurstu.2010.06.004>
- Posey, C., Roberts, T., Lowry, P. B., Courtney, J., & Bennett, B. (2011). Motivating the insider to protect organizational information assets: Evidence from protection motivation theory and rival explanations. The Dewald Roode workshop in information systems security, Blacksburg, Virginia, USA.
- Press room European Commission, E.-i. (2014, 2014-09-10). Reuse is the key to the circular economy. *Eco-innovation at the heart of European policies*. https://ec.europa.eu/environment/ecoap/about-eco-innovation/experts-interviews/reuse-is-the-key-to-the-circular-economy_en
- Preston, F., Lehne, J., & Wellesley, L. (2019). An inclusive circular economy: Priorities for developing countries. *The Royal Institute of International Affairs: Chatham House*, 1-82.
- Prieto-Diaz, R. (1985). *A software classification scheme (reusability, libraries, development)* (Publication Number 8527430) University of California]. Irvine, CA.
- Prieto-Diaz, R. (2003). A faceted approach to building ontologies. Proceedings Fifth IEEE Workshop on Mobile Computing Systems and Applications,
- Prieto-Diaz, R., & Freeman, P. (1987). Classifying software for reusability. *IEEE Software*, 4(1), 6-16. <https://doi.org/10.1109/ms.1987.229789>
- Prieto-Díaz, R. (1990). Domain analysis. *ACM SIGSOFT Software Engineering Notes*, 15(2), 47-54. <https://doi.org/10.1145/382296.382703>
- Prieto-Díaz, R. (1991a). Implementing faceted classification for software reuse. *Communications of the ACM*, 34(5), 88-97. <https://doi.org/10.1145/103167.103176>
- Prieto-Díaz, R. (1991b). Making software reuse work: An implementation model. *ACM SIGSOFT Software Engineering Notes*, 16(3), 61-68. <https://doi.org/10.1145/127099.127107>
- Prieto-Díaz, R. (2004). Requirements engineering in the information assurance domain: The common criteria evaluation process. In J. C. S. do Prado Leite & J. H. Doorn (Eds.), *Perspectives on software requirements* (pp. 139-167). The Springer International Series in Engineering and Computer Science. https://doi.org/10.1007/978-1-4615-0465-8_7
- Prusak, L., & Davenport, T. (1998). Working knowledge: How organizations manage what they know. *Ubiquity*, 1-15.
- Ram, B., & Devi, A. (2019). Code reuse & reusability of the software. *International Research Journal of Engineering and Technology (IRJET)*, 6(5), 1337-1350.
- Ram, K. (2013). Git can facilitate greater reproducibility and increased transparency in science. *Source Code for Biology and Medicine*, 8(1), 7. <https://doi.org/10.1186/1751-0473-8-7>
- Rasche, C., & Seisreiner, A. (2018). Guidelines for business case analysis. 1-12. Retrieved 2021-04-11, from <https://www.uni->

potsdam.de/fileadmin/projects/professional-services/downloads/skripte-ss/Anleitung_Case_Studies.pdf

- Rashid, Y., Rashid, A., Warraich, M. A., Sabir, S. S., & Waseem, A. (2019). Case study method: A step-by-step guide for business researchers. *International Journal of Qualitative Methods*, 18, 1-13. <https://doi.org/10.1177/1609406919862424>
- Reddy, G. S., & Jalender, B. (2011). Strategies for deploying reusable software components. *International Journal of Graphics & Image Processing*, 2(4), 264-273.
- Redman-Maclaren, M. L., Api, U. K., Darius, M., Tommbe, R., Mafile'O, T. A., & Maclaren, D. J. (2014). Co-interviewing across gender and culture: Expanding qualitative research methods in Melanesia. *BMC Public Health*, 14(1), 1-7. <https://doi.org/10.1186/1471-2458-14-922>
- Reinhartz-Berger, I. (2024). Challenges in software model reuse: cross application domain vs. cross modeling paradigm. *Empirical software engineering : an international journal*, 29(1), 16. <https://doi.org/10.1007/s10664-023-10386-9>
- Rider, C., Hong, L., O'Connell, A., Stewart, M., & Herring, M. (2006). *Taxation problems in the commercialisation of intellectual property*.
- Rikap, C., & Lundvall, B.-Å. (2020). Big tech, knowledge predation and the implications for development. *Innovation and Development*, 1-28. <https://doi.org/10.1080/2157930x.2020.1855825>
- Ritter, H. (1838). *The history of ancient philosophy* (Vol. 4). Henry G. Bohn.
- Rodríguez García, J. M. (2001). Scientia potestas est knowledge is power: Francis Bacon to Michel Foucault. *Neohelicon*, 28(1), 109-121. <https://doi.org/10.1023/a:1011901104984>
- Romer, P. M. (1990). Endogenous Technological Change. *Journal of Political Economy*, 98(5), S71-S102. <https://doi.org/10.1086/261725>
- Rorty, R. (1982). *Consequences of pragmatism: Essays, 1972-1980* (Vol. 42). University of Minnesota Press. <https://doi.org/10.1515/9781400848393-043>
- Rose, A. (2021). Hard lessons from the sharing economy. *Academia Letters*, April 2021, 6. <https://doi.org/https://doi.org/10.20935/AL597>
- Rose, C. (2019). *Systems for reuse, repurposing and upcycling of existing building components* University College London]. London, UK.
- Rosenbaum, P. R. (2002). Overt bias in observational studies. In *Observational Studies* (2nd ed., pp. 71-104). Springer. https://doi.org/10.1007/978-1-4757-3692-2_3
- Roth, W.-M., & von Unger, H. (2018). Current perspectives on research ethics in qualitative research. *Forum qualitative social research (FQS)*, 19(3), 1-12. <https://doi.org/10.17169/fqs-19.3.3155>

- Rowley, J. (2007). The wisdom hierarchy: Representations of the DIKW hierarchy. *Journal of Information Science*, 33(2), 163-180. <https://doi.org/10.1177/0165551506070706>
- Roztock, N., Soja, P., & Weistroffer, H. R. (2019). The role of information and communication technologies in socioeconomic development: Towards a multi-dimensional framework. *Information Technology for Development*, 25(2), 171-183. <https://doi.org/10.1080/02681102.2019.1596654>
- Rubén, D.-H. (1979). *Marxism and materialism - a study in Marxist theory of knowledge* (2nd. ed.). Harvester Press.
- Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2), 131-164. <https://doi.org/10.1007/s10664-008-9102-8>
- Sahibzada, U. F., Cai, J., Latif, K. F., & Sahibzada, H. F. (2019). Knowledge management processes, knowledge worker satisfaction, and organizational performance: Symmetric and asymmetrical analysis. *Aslib journal of information management*, 72(1), 112-129. <https://doi.org/10.1108/AJIM-10-2019-0276>
- Salesmate. (2018). 7 sales pipeline stages in CRM that make your small business a success. *Salesmate*. Retrieved 2021-04-11, from <https://salesmateio.medium.com/7-sales-pipeline-stages-in-crm-for-your-successful-small-business-e8bd25d7498c>
- Salgot, M., Oron, G., Cirelli, G., Dalezios, N., Diaz, A., & Angelakis, A. (2017). Criteria for wastewater treatment and reuse under water scarcity. In S. Eslamian & F. A. Eslamian (Eds.), *Handbook of drought and water scarcity* (pp. 263-282). CRC Press. <https://doi.org/10.1201/9781315226781-15>.
- Sandhu, A. K., & Batth, R. S. (2021a, 2021-01-19). Integration of artificial intelligence into software reuse: An overview of software intelligence. 2nd International Conference on Computation, Automation and Knowledge Management (ICCAKM), Noida, India.
- Sandhu, A. K., & Batth, R. S. (2021b). Software reuse analytics using integrated random forest and gradient boosting machine learning algorithm. *Software: Practice and Experience*, 51(4), 735-747. <https://doi.org/10.1002/spe.2921>
- Saunders, M. N. K., Thornhill, A., & Lewis, P. (2019). Understanding research philosophy and approaches to theory development. In *Research Methods for Business Students* (8th ed., pp. 128-170). Pearson Education Limited.
- Schloss, P. D. (2018). Identifying and overcoming threats to reproducibility, replicability, robustness, and generalizability in microbiome research. *MBio*, 9(3), 1-13. <https://doi.org/10.1128/mBio.00525-18>
- Schultz, J. R. (2022). Perceptions about reality undermine collective thinking. *Global Business and Organizational Excellence*, 41(2), 35-42. <https://doi.org/10.1002/joe.22140>
- Schumpeter, J. A. (2017). *The theory of economic development: An inquiry into profits, capital, credit, interest, and the business cycle*. Routledge.
- Schwartz, D. G. (2005). *Encyclopedia of knowledge management*. IGI Global.

- Schwier, R. A., Campbell, K., & Kenny, R. (2004). Instructional designers' observations about identity, communities of practice and change agency. *Australasian Journal of Educational Technology*, 20(1), 69-100. <https://doi.org/10.14742/ajet.1368>
- Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3), 379-423.
- Shen, Z., & Spruit, M. (2019). A systematic review of open source clinical software on GitHub for improving software reuse in smart healthcare. *Applied Sciences*, 9(1), 150. <https://doi.org/10.3390/app9010150>
- Shiva, S. G., & Shala, L. A. (2007, 2007-04-01). Software reuse: Research and practice. Fourth International Conference on Information Technology (ITNG'07), Las Vegas, NV.
- Sidmou, H. (2021). Frugal consumption, an alternative in times of crisis? A reflection on the responsible factors. *Academia Letters*, 1-5. <https://doi.org/10.20935/AL743>
- Sinha, S., & Modak, N. M. (2021). A systematic review in recycling/reusing/re-manufacturing supply chain research: A tertiary study. *International Journal of Sustainable Engineering*, 14(6), 1411-1432. <https://doi.org/10.1080/19397038.2021.1986594>
- Sodhi, G. S., & Rattan, D. (2021). An insight on software features supporting software transplantation: A systematic review. *Archives of Computational Methods in Engineering*. <https://doi.org/10.1007/s11831-021-09593-8>
- Song, S., Nerur, S., & Teng, J. T. C. (2007). An exploratory study on the roles of network structure and knowledge processing orientation in work unit knowledge management. *ACM SIGMIS Database: the DATABASE for Advances in Information Systems*, 38(2), 8-26. <https://doi.org/10.1145/1240616.1240620>
- Spender, J.-C. (2008). Organizational learning and knowledge management: Whence and whither? *Management Learning*, 39(2), 159-176. <https://doi.org/10.1177/1350507607087582>
- Spilhaus, A. (1971). The next industrial revolution. *Proceedings of the American Philosophical Society*, 115(4), 324-327.
- Spoelstra, W., Iacob, M., & Van Sinderen, M. (2011, 2011-01-01). Software reuse in agile development organizations - a conceptual management tool. Proceedings of the 2011 ACM Symposium on Applied Computing - SAC '11, TaiChung, Taiwan.
- Sridhar, M. B., Srinivas, Y., & Krishna Prasad, M. H. M. (2012). Software reuse in cardiology related medical database using K-Means clustering technique. *Journal of Software Engineering and Applications*, 5(9), 682-686. <https://doi.org/10.4236/jsea.2012.59081>
- Stahel, W. R. (2016). Circular economy. *Nature*, 531, 435-438. <https://doi.org/10.1038/531435a>
- Stahel, W. R., & MacArthur, E. (2019). *The circular economy: A user's guide*. Routledge. <https://doi.org/10.4324/9780429259203>

- Statistics, B. o. L. (2020). *Employee tenure in 2020* (BLS News Release - Bureau of Labor Statistics, Issue. <https://www.bls.gov/news.release/pdf/tenure.pdf>)
- Stein, E. W., & Zwass, V. (1995). Actualizing organizational memory with information systems. *Information systems research*, 6(2), 85-117. <https://doi.org/10.1287/isre.6.2.85>
- Stenholm, D., Corin Stig, D., Ivansen, L., & Bergsjö, D. (2019). A framework of practices supporting the reuse of technological knowledge. *Environment Systems and Decisions*, 39(2), 128-145. <https://doi.org/10.1007/s10669-019-09732-4>
- Stephens, K. K. (2012). Multiple Conversations During Organizational Meetings. *Management Communication Quarterly*, 26(2), 195-223. <https://doi.org/10.1177/0893318911431802>
- Stuckey, H. L. (2013). Three types of interviews: Qualitative research methods in social health. *Journal of Social Health and Diabetes*, 1(2), 56-59. <https://doi.org/10.4103/2321-0656.115294>
- Succar, B. (2013). *BIM Performance & Measurement Improvement* [PDF of a Powerpoint presentation]. Sustainable Design through Building Information Modelling, Kuala Lumpur, University of Malaya. https://www.academia.edu/3529511/BIM_Performance_Measurement_and_Improvement_presentation_slides_sm=b?source=news_feed_share
- Succar, B., & Poirier, E. (2020). Lifecycle information transformation and exchange for delivering and managing digital and physical assets. *Automation in Construction*, 112, 1-22. <https://doi.org/10.1016/j.autcon.2020.103090>
- Svahnberg, M., & Gorschek, T. (2017). A model for assessing and re-assessing the value of software reuse. *Journal of Software: Evolution and Process*, 29(4), 1-16. <https://doi.org/10.1002/smr.1806>
- Talwar, S., Talwar, M., Kaur, P., & Dhir, A. (2020). Consumers' Resistance to Digital Innovations: A Systematic Review and Framework Development. *Australasian Marketing Journal*, 28(4), 286-299. <https://doi.org/10.1016/j.ausmj.2020.06.014>
- Taraszewski, S. A. (2017). *Understanding knowledge storage/retrieval system success: An analytic network process perspective* Cleveland State University]. Cleveland, OH.
- Teerajetgul, W., & Charoenngam, C. (2006). Factors inducing knowledge creation: Empirical evidence from Thai construction projects. *Engineering, Construction and Architectural Management*, 13(6), 584-599. <https://doi.org/10.1108/09699980610712382>
- Tenopir, C., Rice, N. M., Allard, S., Baird, L., Borycz, J., Christian, L., . . . Sandusky, R. J. (2020). Data sharing, management, use, and reuse: Practices and perceptions of scientists worldwide. *Plos One*, 15(3), 1-26. <https://doi.org/10.1371/journal.pone.0229003>
- Terjesen, S. A. (2003). Knowledge Management at Accenture: 1992 - January 2001. In P. Gooderham & O. Nordhaug (Eds.), *International management: Cross-boundary challenges* (pp. 234-257). Blackwell Publishing.

- Tidd, J., & Bessant, J. R. (2020). *Managing innovation: Integrating technological, market and organizational change*. Wiley.
- Tomo, O. (2020). Intangible Asset Market Value Study. *Intangible Asset Market Value Study*. Retrieved 2021-05-02, from <https://www.oceantomo.com/intangible-asset-market-value-study/>
- Toxboe, A. (2021). Making the Fogg behavior model actionable. *UI Patterns*. <http://ui-patterns.com/blog/making-the-fogg-behavior-model-actionable>
- Tripathy, P., & Naik, K. (2014). *Software evolution and maintenance: A practitioner's approach*. John Wiley & Sons.
- Trottnow, J., Greenly, W., Shaw, C., Hudson, S., Helzle, V., Vera, H., & Ring, D. (2020). SAUCE: Asset libraries of the future. The Digital Production Symposium, online.
- Ueki, H., Ueki, M., Linowes, R., & Mroczkowski, T. (2011). A comparative study of enablers of knowledge creation in Japan and US-based firms. *Asian Business & Management*, 10(1), 113-132. <https://doi.org/10.1057/abm.2010.33>
- Upadhyay, P., & Kumar, A. (2020). The intermediating role of organizational culture and internal analytical knowledge between the capability of big data analytics and a firm's performance. *International Journal of Information Management*, 52, 1-16. <https://doi.org/10.1016/j.ijinfomgt.2020.102100>
- Van Buren, N., Demmers, M., Van Der Heijden, R., & Witlox, F. (2016). Towards a circular economy: The role of Dutch logistics industries and governments. *Sustainability*, 8(7), 1-17. <https://doi.org/10.3390/su8070647>
- Vasanth, G., Corney, J., Stuart, S., Sherlock, A., Quigley, J., & Purves, D. (2020). A probabilistic design reuse index for engineering designs. *Journal of Mechanical Design*, 142(10), 1-11. <https://doi.org/10.1115/1.4046435>
- Vasiljuk, D., & Budke, A. (2021). Multiperspectivity as a process of understanding and reflection: Introduction to a model for perspective-taking in Geography education. *European Journal of Investigation in Health, Psychology and Education*, 11(2), 529-545. <https://doi.org/10.3390/ejihpe11020038>
- Vedres, B. (2022). Network mechanisms in innovation: borrowing and sparking ideas around structural holes. In *Handbook of Sociological Science* (pp. 423-442). Edward Elgar Publishing. <https://doi.org/10.4337/9781789909432.00031>
- Vizcaino, N. (2017). *Software evolution: hypergraph based model of solution space and meta-search* [University of Reading]. Reading, UK.
- Vogl, S., Zartler, U., Schmidt, E.-M., & Rieder, I. (2018). Developing an analytical framework for multiple perspective, qualitative longitudinal interviews (MPQLI). *International Journal of Social Research Methodology*, 21(2), 177-190. <https://doi.org/10.1080/13645579.2017.1345149>
- Vom Brocke, J., Simons, A., Riemer, K., Niehaves, B., Plattfaut, R., & Cleven, A. (2015). Standing on the shoulders of giants: Challenges and recommendations of literature search in information systems research. *Communications of the Association for Information Systems*, 37, 1-9. <https://doi.org/10.17705/1cais.03709>

- Vrzakova, H., Amon, M. J., Rees, M., Faber, M., & D'Mello, S. (2021). Looking for a deal? *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW3), 1-35. <https://doi.org/10.1145/3434169>
- Wallace, G. W. (2018, 2018-01-14). L&D: Push-pull knowledge management systems – Part 1. *Pursuing Performance - Enterprise Process Performance Improvement Consultancy*. <https://eppic.biz/2018/01/14/ld-push-pull-knowledge-management-systems-part-1/>
- Wang, J., & Yang, J. (2015). An empirical study of employees' tacit knowledge sharing behavior. *Journal of Systems Science and Information*, 3(3), 264-278. <https://doi.org/doi:10.1515/JSSI-2015-0264>
- Warren, K. (2020). Qualitative data analysis methods 101 - the big 5 methods and examples. (May 2020). Retrieved 2021-09-11, from <https://gradcoach.com/qualitative-data-analysis-methods/>
- Wegner, C. (2016). Upcycling. In V. P. Glăveanu, L. Tanggaard Pedersen, & C. Wegner (Eds.), *Creativity - A New Vocabulary* (1st ed., pp. 181-188). Palgrave Macmillan. <https://doi.org/10.1057/9781137511805>
- Weissenberger-Eibl, M. A., & Hampel, T. (2021). What do we have in-common? Overcoming the not-invented-here syndrome through recategorisation. *International journal of innovation management*, 25(6), 2150070-2150071 - 2150070-2150033. <https://doi.org/10.1142/S1363919621500705>
- Weller, S. (2017). Using internet video calls in qualitative (longitudinal) interviews: Some implications for rapport. *International Journal of Social Research Methodology*, 20(6), 613-625. <https://doi.org/10.1080/13645579.2016.1269505>
- Wentzel, K. D. (1994). Software reuse - facts and myths. Proceedings of 16th International Conference on Software Engineering, Sorrento, Italy.
- Whalen, K. A., Milios, L., & Nussholz, J. (2018). Bridging the gap: Barriers and potential for scaling reuse practices in the Swedish ICT sector. *Resources, Conservation and Recycling*, 135, 123-131. <https://doi.org/http://dx.doi.org/10.1016/j.resconrec.2017.07.029>
- Wickramasinghe, N., Gupta, J. N. D., & Sharma, S. K. (2005). *Creating knowledge-based healthcare organizations*. Idea Group Publishing.
- Wijnhoven, F. (2008). Manufacturing knowledge work: The European perspective. In A. Bernard & S. Tichkiewitch (Eds.), *Methods and tools for effective knowledge life-cycle-management* (pp. 23-44). Springer. https://doi.org/10.1007/978-3-540-78431-9_2
- Wikfeldt, E. (1993). Generalising from case studies. (January 1993). <https://doi.org/10.13140/RG.2.2.25554.76480>
- Wilson, R. M. S., Stenson, J., & Oppenheim, C. (2000). *Valuation of information assets*.
- WIPO. (2022). *Patents*. World Intellectual Property Organization. Retrieved 2022-07-12 from <https://www.wipo.int/patents/en/index.html>

- Woerner, S. L., Weill, P., & McDonald, M. P. (2013). Turn time into money: Faster growth through digital reuse. *European Business Review*, 25(3), 38-42.
- Wolfram, K., Martine, M., & Sudnik, P. (2020). Recognising the types of software assets and its impact on asset reuse. European Conference on Software Process Improvement (EuroSPI), Düsseldorf, Germany.
- Wright, L. (1973). Rival Explanations. *Mind*, 82(328), 497-515.
<https://doi.org/10.1093/mind/LXXXII.328.497>
- Yeh, Y. J., Lai, S. Q., & Ho, C. T. (2006). Knowledge management enablers: A case study. *Industrial Management & Data Systems*, 106(6), 793-810.
<https://doi.org/10.1108/02635570610671489>
- Yin, R. K. (2018). *Case study research and applications* (6th ed.). SAGE.
- Yoo, S.-J., Sawyerr, O., & Tan, W.-L. (2016). The mediating effect of absorptive capacity and relational capital in alliance learning of SMEs. *Journal of Small Business Management*, 54, 234-255. <https://doi.org/10.1111/jsbm.12299>
- Yoshida, E. (2014). Efforts to reuse software assets. *NTT Technical Review*, 1-4. Retrieved 2021-06-12, from <https://www.ntt-review.jp/archive/ntttechnical.php?contents=ntr201412fa4.html>
- Youndt, M. A., Subramaniam, M., & Snell, S. A. (2004). Intellectual capital profiles: An examination of investments and returns. *Journal of Management Studies*, 41(2), 335-361. <https://doi.org/10.1111/j.1467-6486.2004.00435.x>
- Younoussi, S., el Rhaffari, I., Amoud, M., & Roudies, O. (2019). Software reuse in organizations: A survey in Moroccan software industry context. *Journal of Software*, 14(4), 153-167. <https://doi.org/10.17706/jsw.14>. 4 . 153 - 167
- Zabardast, E., Frattini, J., Gonzalez-Huerta, J., Mendez, D., Gorschek, T., & Wnuk, K. (2022). Assets in Software Engineering: What are they after all? *Journal of Systems and Software*, 111485.
- Zabardast, E., Gonzalez-Huerta, J., Gorschek, T., Šmite, D., Alégroth, E., & Fagerholm, F. (2023). A taxonomy of assets for the development of software-intensive products and services. *The Journal of systems and software*, 202, 1-23. <https://doi.org/https://doi/10.1016/j.jss.2023.111701>
- Zand, M., Basili, V., Baxter, I., Griss, M., Karlsson, E.-A., & Perry, D. (1999, 1999-01-01). Reuse R&D: Gap between theory and practice. Proceedings of the 1999 symposium on software reusability - SSR '99, Los Angeles, CA.
- Zand, M., Bassett, P., & Prieto-Díaz, R. (2001, 2001-01-01). Closing panel (panel session): where are we standing? can we say "reuse is dead, long live reuse" or is it too soon? Proceedings of the 2001 symposium on Software reusability putting software reuse in context - SSR '01, Toronto, Canada.
- Zechmann, A. (2016). *Assessing the economic value of data assets* (Data Management Publications - Data Governance & Data Management, Issue. C. C. C. D. Q. C. CDQ). https://www.cc-cdq.ch/system/files/Data_Valuation-Work_Report_2016%20-%20final_0.pdf

- Zhou, J. (2001). *Applying Concepts of Software Reuse to the Implementation of Data Warehouse ETL Systems*.
<https://www.coursehero.com/file/66205337/DA-Zhou-Nov-2001doc/>
- Zhou, X. (2019). A review of complementary assets. *American Journal of Industrial and Business Management*, 09(09), 1772-1780.
<https://doi.org/10.4236/ajibm.2019.99116>
- Zhou, Z., & Pazos, P. (2020). Empirical perspectives of transactive memory systems: A meta-analysis. *Team performance management*, 26(7/8), 409-427. <https://doi.org/10.1108/TPM-05-2020-0036>
- Zhu, Z. (2009, 2009-07-01). Study and Application of Patterns in Software Reuse. 2009 IITA International Conference on Control, Automation and Systems Engineering (case 2009),
- Zorzetti, M., Signoretti, I., Salerno, L., Marczak, S., & Bastos, R. (2022). Improving agile software development using user-centered design and lean startup. *Information and Software Technology*, 141, 1-14.
<https://doi.org/10.1016/j.infsof.2021.106718>
- Zozas, I., Ampatzoglou, A., Bibi, S., Chatzigeorgiou, A., Avgeriou, P., & Stamelos, I. (2019). REI: An integrated measure for software reusability. *Journal of Software: Evolution and Process*, 31(8), 1-18.
<https://doi.org/10.1002/smr.2216>

Appendices

Appendix A: Operational Knowledge Processes

This appendix illustrates how the researcher picked up knowledge processes from the literature. It became apparent that the best way to classify the different knowledge processes would be into “knowledge creation”, “knowledge storage”, “knowledge distribution” and “knowledge application”.

Table 5.1 Overview of how knowledge processes were bundled (source: author)

Song, Nerur & Teng, 2007	Spender, 2008	Wijnhoven, 2008	Fogg, 2009 Persuasive Behaviour	Eid & Nuhu, 2011	Burciu & Kicsi, 2015	Garcia-Perez, Ayres, 2009, 2015	Taraszewski, 2017	Manesh et al., 2021	Source/ KM sub-process
									Trigger for retrieval of existing knowledge
									Knowledge discovery
		Knowledge creation		Knowledge creation	Knowledge	Knowledge acquisition		Knowledge	Creation
Creation	Indentification				Knowledge creation				Identification
									Creation
									Capturing
									Organising
	Codifying								Coding
									Representation
Storage		Knowledge retrieval, storage, removal		Knowledge sharing		Knowledge conversion	Knowledge storage, retrieval	Knowledge documentation	Storage
									Retrieval
		configuring			Knowledge Maintenance				Manipulation
Access	sharing								Access
									Sharing
Dissemination				Knowledge dissemination				Knowledge transfer/sharing	Distribution
									Transformation
									Modelling
									Evaluation
									Interpretation
									Exploitation
						Knowledge application			Application
		Sharing							Embedding
									Reusing
									Update
						Protection of			Protection
	Establishing & retaining the ownership								Organisational memory
									Ownership

	Huber Jr, 1991	Devanbu et al., 1991	Hedlund, 1994	Nonsaka, 1994	Prusak & Davenport, 1998	Alavi & Leidner, 2001	Kneuper, 2002	Schwier, Campbell, and Kenny, 2004	Schwartz, 2005
CREATION of knowledge									
	Knowledge	Discovery task		Creation			Knowledge identification		Knowledge Discovery via Combination + Socialisation
STORAGE of knowledge incl. maintenance, updating		Discovery task	Generation		Creation	Acquisition	Knowledge acquisition		Knowledge acquisition
					Capturing	Identification	Knowledge development		Creation
					Capturing	Construction			Knowledge capturing
						Capturing			Mapping, indexing
			Representation			coding		Coding of information	
DISTRIBUTION of knowledge (incl. retrieval, transfer and sharing)			Storage	Storage		storage		Storage of information	Knowledge storage
						manipulation			Knowledge retrieval
					Sharing	Sharing, Leveraging			Knowledge sharing
	Knowledge distribution		Transfer	Distribution	Distribution	Distribution, Transmission	Knowledge distribution	Distribution of information	Transfer
			Transformation						
									Modelling
	Interpretation of information				Understanding				Evaluation
				Application	Application		Application	Knowledge use	utilisation, application
APPLICATION of knowledge			Embedding						Reuse
									Update
							Knowledge		Protection
	Organisational memory		Projecting of group and organisational			Organisational memory			

Appendix B: Semi-structured Interview Guide

Software Asset Reuse
Knowledge Management Process
Semi-Structured Interview Guide

Kathleen Wolfram



Asset-Knowledge-Creation



Why did you go for an asset?

What motivated you?

How were you incentivised?

Prior to asset creation, did you check for similar assets available?

- How did you check upfront?
- Who or what asked you to check? Trigger?
- Where did you check?
- Who checked?
- Why did you check?

Did similar assets emerge later?

- How did you cope with this?
- Did you try to partner with these assets?
- Are you actively searching for assets to partner with?

When was the asset created?

How long did the asset creation take?

How did you document the asset?

- Did you document “it all”?
- How did you know when to stop?

Who “invented”/created this asset?

How was the knowledge built up?

- By an individual?
- By a group?
- By the corporation?

What would you do differently now – if you created a similar new asset? (Reality vs perfect situation)

Initially, how many features did your asset have?

- How many features does your asset have today?
- How did new features emerge over time? In the early years? Constantly over time?

Did you share features you developed for a new client with your already existing clients?

Do you have a development plan for your asset?

- Who owns it?
- Why do you need it?
- How much did it help you in the past?
- What are key elements on the plan for the asset now?

What kind of knowledge is contained in your asset?

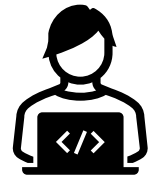
- Coding knowledge (technology dependent or independent)?
- Business knowledge?
- Other knowledge?

How has that knowledge been gathered? Through learning?

- How was the learning for the asset done?
- Who did the learning?
- Were there any milestones? Key events? E.g cloud

- What are aspects that are by now “forgotten”?
- Has the asset knowledge enlarged its scope?
- Did it start as the knowledge of an individual?
 - Did it then become the knowledge of a team?
 - Do you now see it as the knowledge of our organisation? Why?
- Who was your first client?
- Did your client set change over time?
- How did the asset evolve over time?
- Content/functions
 - Technology
- How much “old” asset still exists? (Still some lines of code from day 1?)
- Could we systematically reuse your asset?
- How would this look? What could systematic reuse mean?
 - Could you provide numbers on this?
- How to achieve systematic reuse?
- What makes your asset fit for reuse?
 - What could be further improved in the future?
 - How many of all potential opportunities are you serving with this asset?
- Did you contact the offering lead?
- When? Prior to asset creation?
 - Why? To avoid any duplication?
- Did you map your asset against an offering?
- When?
 - Against how many offerings?
 - What were the benefits?
 - Did your mapping change over time?
- How can you describe the value of the knowledge/asset to the business of our organisation?
- What impact will our move to cloud have on the way we create assets?
- Do you own similar assets that are very close to this asset? – Why?

Asset-Knowledge-Storage



Where did you first store the asset?

- in the asset sharing platform?
- in GitHub

Where do you now store the asset?

What is your experience storing the asset?

How do you keep an overview over your many, many versions?

What exactly did you store about the asset? – There must be LOT of information given all its many features ...

- Are you storing every kind of code this way?
- Or you only store asset code this way?

Who is your audience? Who will benefit from the stored asset?

- Other teams?
- Only your own team?
- How often do other teams spot the asset in the repository and then reach out to you?

If transition occurs and you change the underlying technology of your asset ... how does this impact your way of storing the asset?

When did you store the asset?

Why did you store the asset?

How do you keep an overview over all the asset versions ... all the asset knowledge? Is there a trick or a lesson learned?

Are you the one who has the best overview over the asset knowledge?

What would happen if you were not available for some time (Covid-19, sabbatical, retirement, birth of a child)? – How could the project carry on?

- How could the asset team carry on with you as the expert not being there? Is there a stronger dependency on the storage or on the expert person?
- To what extent would they depend on what is written down?
- Or could the absence of the key expert be compensated for by a backup person?
- Are you using any AI to capture knowledge on the asset?

Who helped you to store the asset?

Who has (editor) access to the stored asset?

Who told you to store the asset?

What motivated you?

How are you incentivised to store the asset?

What does systematic asset reuse mean to you? How would you explain that in your own words?

- Is this something you try to achieve?
- How would systematic reuse look? Can you explain that in your own words?
- How do you achieve systematic reuse from a storage point of view?

How often do you update the asset record in the repository?

Can you see innovation resulting from the reuse of your asset? Have you been in situations where your asset ignited a magic spark to lead to the emergence of new, better approaches ... that would not have happened if your asset had not been there?

The knowledge we talk about here is an asset. Do you think there is a difference in managing the knowledge of an asset vs the knowledge of a product?

How many patents have been filed related to the asset?

Why were patents filed?

Did the asset team draw benefit from any of the patents?

- How?

- Did you go to court?

Would you see a patent as knowledge storage?

How do you store captured business knowledge?

- Only in the code of the asset?

- Or do you have specific process descriptions or similar?

How do you organise the (business and coding) knowledge?

- Do you tag/classify the knowledge?

How did you store the asset knowledge in the past vs today vs in the future?

- Is there a trend?

- Anything new coming up?

Are you familiar with ServiceNow? Do you think a tool like this or something similar would help knowing where each software is installed?

What are your main reasons for bringing your asset on the cloud?

- What will change compared to on premises?

- What new advantage can you foresee for the future?

How do you benefit from using automation?

Any challenges?

Any process improvement ideas?

Do you have a cool story to share that would be typical or illustrative of the process?

Was there a major failure that happened as part of the process?

- What impact did it have on business?

- What did the asset team learn from it?

Is there any aspect about this process which you think is important that you have not yet touched on?

Do you see a trend for this process? – Is it gaining importance? Is it shifting towards a certain topic?

Who sponsored the process/process outcome? Why?

Do you have any ongoing, unresolved problem with this process?

Did you find some unique solutions/approaches to this process in the past?

What are important metrics about the process?

What is the process before this process?

What is the process after this process?

How did your role in the team change over time?

To what extent did you accumulate not only asset knowledge but also related business knowledge?

Asset-Knowledge-Distribution



What motivates you to distribute the asset?

How are you incentivised for asset distribution/marketing?

What kind of targets do you have and how much does this influence you in distributing the asset?

What would be the maximum possible reuse of the asset?

- How close to this are you with this asset? Why?

How do you achieve systematic reuse?

How many opportunities for your assets are in our system? How many of these do you think are currently covered by the asset?

- Why is there a discrepancy?

- What can be done to reduce the discrepancy?

What do you do to promote your asset and how successful are these approaches according to your experience?

- Industry sessions

- Service line sessions

- Community work

- Opportunity scans in Atlas ... if yes how often do you scan and how did you come up with the relevant keywords for your search?

- Personal network

- Run education sessions

Looking back at your history of wins: Who triggered the reuse? – the asset owning team or the opportunity owning team? – What's the ratio?

Looking at your pipeline – what's the ratio of new cases vs extension?

How many clients did you have over time?

How did your asset revenue develop over time? Constantly rising? Rather flat? Slightly declining.

- What's the reason?

- What can be done to improve the situation?

Do you think this amount of revenue could have been achieved alternatively?

What is the innovation that occurred as part of the reuse?

Would you agree that to achieve reuse, we need to have “the right asset at the right time”?

- When is the earliest time to include this asset in an opportunity?

- When is the latest time to include this asset in an opportunity?

Did this particular asset reuse generate signings/revenue/savings?

- How and to whom was this reuse reported?

- How important are previous instances of reuse/references for your future reuse? E.g., we are sharing the reuse on the asset sharing platform – is this helping you?

- Where is the reuse now listed for further reference?

Do you already have SaaS clients for your asset? How do they locate the asset?

What changes with SaaS from a marketing point of view?

Can you give examples of when innovative ideas opened up new use cases and clients for your asset?

- How did that innovation happen?
- What triggered this innovation?

How did the use cases of the asset change over time?

Internal Marketing

How do you think about exec sponsorship of e.g. a global and local industry lead for your asset? What effect did it have on the reuse of your asset?

How closely do you work with the global industry Centre of Competence? – What effect did this have on the reuse?

How closely do you interact with members of the industry community? What effect does this have on the reuse of your asset?

How do you make the asset info available via community and / or wiki? – What effect did it have on the reuse of your asset?

When did you last produce and make available a sub-titled video on the asset e.g. on YouTube? What effect did this have?

What do you think about running internal education sessions on the asset and inviting our key sellers to these? What effect did this have?

Did you talk to the company's key sellers 1:1 about the asset? What effect did this have?

What do you think about mentioning your asset again and again in internal newsletters? What effect did this have?

How did you experiment with specific structures of the asset price (low budget PoC, T-shirt price)?

What is your experience with providing, documenting and advertising the support structure for the asset (tier 3 support)? Does it help to increase the reuse of the asset?

Who is part of the asset owning team and why?

Responsible for

- Selling the asset directly to clients in that industry; in permanent and close connection with client execs
- Constantly maintaining the asset's marketing material
- Develop, track and push the asset pipeline
- Drive asset commercialisation

How closely are you working with the respective offering owners who manage the offerings your asset is part of? What impact does this have?

External Marketing

How did you as the asset owner benefit from being an (active) member of related external associations or organisations where clients are represented (to attend meetings / shows, give presentations)? Was there a stage to talk about the asset?

How does your team feed the public with info on the asset via Twitter, blog, external community, external asset website?

Where did you upload a video on YouTube that explains the asset in action (even if it is not yet fully developed)?

How do you attend international fairs, shows and meet with the clients?

Ask potential clients what they have currently installed, what their pain points are, their budget is, etc

Log the details of potential clients and follow up

How do you organise special meetings / workshops / seminars with external clients?

How do you stay in touch with the clients, develop a roadmap for their individual asset roll-out (starting with a light / basic version)?

How do you have the asset installed @ the company's client centres and work actively with the client centre teams, with Front Office Digitisation and Integrated Smarter Solutions Team to drive visitors to come and see the asset, track the meetings and feedback received

How do you apply for internal and external awards for this asset? What effect does it have on the asset revenue?

Do you regularly issue an asset solution newsletter to external clients AND internal team?

How do you involve sales & marketing e.g. for the production of flyers, newsletters?

Any challenges?

Any process improvement ideas?

Do you have a cool story to share that would be typical or illustrative of the process?

Was there a major failure that happened as part of the process?

- What impact did it have on business?

- What did the asset team learn from it?

Is there any aspect about this process which you think is important that you have not yet touched on?

Do you see a trend for this process? – Is it gaining importance? Is it shifting towards a certain topic?

Who sponsored the process/process outcome? Why?

Do you have any ongoing, unresolved problem with this process?

Did you find some unique solutions/approaches to this process in the past?

What are important metrics about the process?

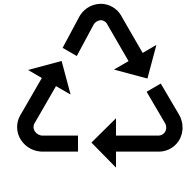
What is the process before this process?

What is the process after this process?

How did your role in the team change over time?

To what extent did you accumulate not only asset knowledge but also the related business knowledge?

Asset-Knowledge-Application



1) Trigger for reuse

What triggered the reuse?
Who triggered “a” reuse?
How was your trigger response?
Why did you respond?
What did you do?
When did you consider reuse?
Who proposed “this” reuse first?
Who else was involved?
Any critical no-reuse moment?
What motivated you?
How were you incentivised?
How to achieve systematic reuse?

2) Asset selection (locating an asset)

How many assets to choose from?
What were asset alternatives?
How was the asset decided on?
What were asset selection criteria?
Have selection criteria been met?
Who decided on the asset?
When was the asset decided on?
How was the decision documented?
What motivated you?
How were you incentivised to carefully select a suitable asset?
How to achieve systematic reuse?
How did you do a requirements analysis?
How did you find the asset via its related offering/solution?
How did you learn about the existence of the asset?
How did you understand the asset?
What did you learn about the asset?
What was the window of decision? (When would it have been too early vs too late in the process?)
- What would have happened if you had heard about the asset circa six months ago?
- What would have happened if you had heard about the asset circa six months later?
- Did this asset arrive at the “right time”?

3) Reusability assessment

Did you have multiple assets to choose from?
What was the input that you used for selecting the asset (e.g., documents from the asset sharing platform)?
How did you assess the asset?
What were asset selection criteria?

Who provided the selection criteria?
Who selected the asset finally?
Was there an asset selection event?
When did the asset get selected (process-wise)?
How important is it for you that the company owns the IP? Would open source be equally acceptable?

4) Asset-Knowledge-Application

How did you retrieve the asset?
Where did you search for the asset?
Where was the asset info stored?
How did you understand the asset?
Who asked you to retrieve it and why?
Who helped you find it and why?
With whom (roles) from the asset owning team did you work to make the reuse happen?
Who (what roles) was in the reusing team?
How big was the reuse team (how many people in total)?
This team reusing the asset was simply a sub-group of the project team, or?
When did you search for the asset?
What motivated you?
How were you incentivised
Contact the asset owner?
- Why?
- When?
Who reused the asset?
How was the asset reused?
What's your reuse experience?
How was the asset owner involved?
How did the asset owner team collaborate with others?
Did the asset keep its promise?
How did the reuse trigger innovation?
Did the asset receive new extensions to make it fit for your client?
What were advantages of reuse?
What were disadvantages of reuse?
Would you reuse again?
Why would you recommend reuse?
What asset metrics did you track?
- Revenue over time?
- Savings over time?
Do you expect a contract expansion?
Do you now pursue further reuses?
Now that you know the asset, are you able to keep on generating further opportunities containing that asset? Why? Why not?

Any challenges?

Any process improvement ideas?

Do you have a cool story to share that would be typical or illustrative of the process?

Was there a major failure that happened as part of the process?

- What impact did it have on business?

- What did the asset team learn from it?

Is there any aspect about this process which you think is important that has not yet touched?

Do you see a trend for this process? – Is it gaining importance? Is it shifting towards a certain topic?

Who sponsored the process/process outcome? Why?

Do you have any ongoing, unresolved problem with this process?

Did you find some unique solutions/approaches to this process in the past?

What are important metrics about the process?

What is the process before this process?

What is the process after this process?

How did your role in the team change over time?

To what extent did you accumulate not only asset knowledge but also the related business knowledge?

How did you feel about the interview?

- Was it stressful or relaxed? Boring or entertaining?

- Too long or did time fly?

- Did you think it was a waste of time or did you gain some new insights for your work?

- Is there anything specific I could improve for the interviews to come?