Understanding the evolutionary notion of BI systems

Since the inception of the field, the notion of *evolutionary* development has been central to the theory and practice of decision-support types of information systems (such as a BI system) (Arnott 2004; Arnott & Pervan 2005). That is, such systems evolve through an *iterative* process of systems design based on dynamic business requirements. It reflects the *adaptive* or *organic* nature of the development of such decision support systems.

Originally, the notion of *evolutionary* development was first suggested by Meador and Ness (1974) and Ness (1975, cited in Arnott & Pervan 2005) as part of their description of middle-out design. Later Courbon, Grajew and Tolovi (1978) put forward the essential statement of evolutionary development, in which they coined the term *'evolutive'* approach. They argue that the implementation processes are not followed in a linear fashion, nor in a parallel mode, but in continuous action cycles that require substantial user involvement. As each 'evolutive' cycle is completed, the system gets closer to its stabilised or matured state. Moreover, the evolutive cycles should be considered as continuous and as rapid as possible in order to meet the ever-changing yet fiercely competitive environment (Courbon, Grajew & Tolovi 1978).

To illustrate, Figure 2.7 depicts the notion of a BI system as an organic cycle that evolves over time. Based on constant evaluation of the information, as well as user feedback, the system resembles a loop that requires re-evaluation of existing BI solutions, and subsequently the system will be modified, optimised and improved accordingly. In other words, completion of the system implementation does not mean that all BI related problems are resolved (Olszak & Ziemba 2007). The system will succeed only when business users keep identifying and modelling knowledge, as well as monitoring and modifying data repositories on an ongoing basis (Olszak & Ziemba 2007). Hence, the entire process is cyclical, but with a series of interrelated steps (Turban *et al*. 2007). In fact, many practitioners follow the depicted model in the process of creating and using business intelligence (Krizan 1999).

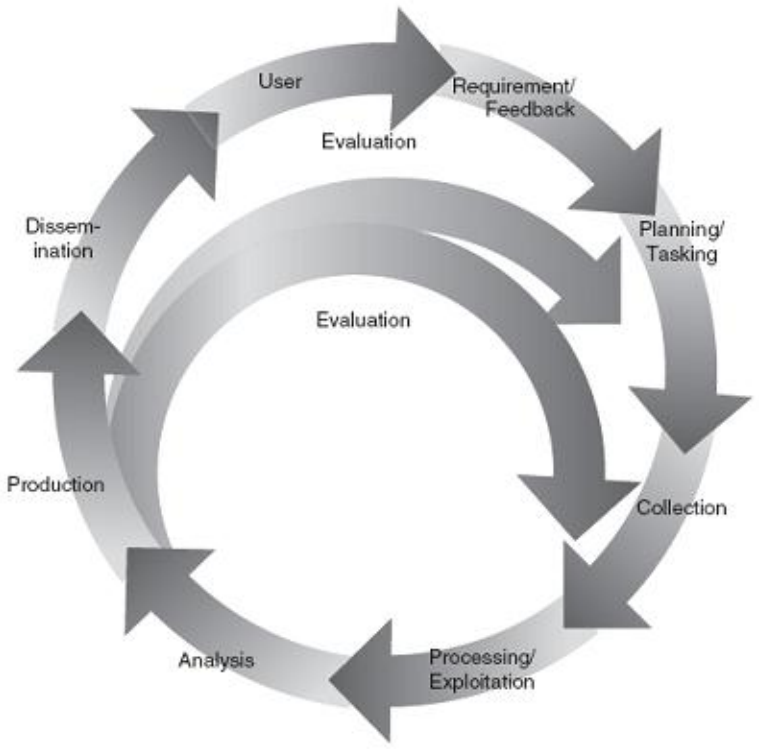

Figure 2.7: Organic process of intelligence creation and use

Source: *Adopted from Krizan (1999); Turban et al. (2007)*

Based on the work of Courbon, Grajew and Tolovi (1978), Keen (1980, cited in Arnott 2004) proposed a framework for understanding the dynamics of the evolution of decision-support systems. The framework was widely known as Keen's *adaptive design* framework, although Arnott (2004) argued that *adaptive development* is a more precise term because the approach incorporates development processes other than design. Since then, this seminal work by Keen (1980) remains the most cited, and thereby the most authoritative, description of the evolutionary approach in decision support systems literature.

Keen's framework consists of three major components: the system analyst, the user, and the system. These components affect each other during the implementation process, forming three major iterative loops, as illustrated in Figure 2.8. Hence the evolution of the matured decision support system is an emergent result of the cyclic operation of the various loops. Based on this use, the user and the system developer learn and plan for the next evolving phase. Thus this evolutionary approach is also best conceived of as a learning process (Courbon, Grajew & Tolovi 1978).

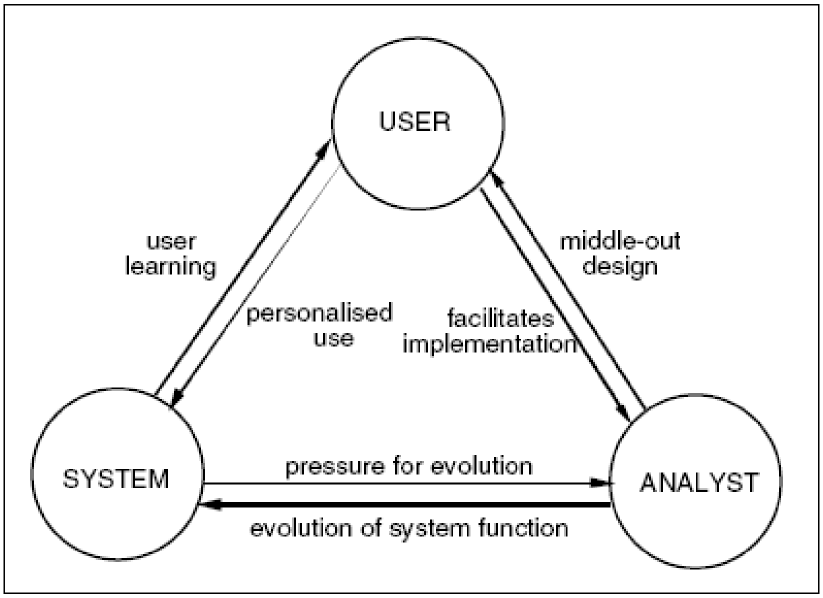

Figure 2.8: Keen's adaptive design framework

Source: *Adopted from Keen (1980, cited in Arnott 2004)*

This assertion was supported by Sprague and Carlson (1982) in their comprehensive study of system adaptation and evolution. They affirmed that such decision-support systems must grow to reach a mature design because no one can anticipate all requirements in advance. The system can never be final but must change dynamically to accommodate changes in information requirements, which are inherently volatile. In addition to the seminal works discussed above, there have been numerous other contributions to the understanding of the evolutionary development, as done by (Keen 1980; Keen & Gambino 1983; Alavi 1984; Young 1989; Arinze 1991; Silver 1991; Suvachittanont, Arnott & O'Donnell 1994).

The primacy of the *evolutionary* approach has been well-recognised within the field of decision support systems. Invariably, a BI system inherits this adaptive nature throughout its implementation lifecycle. To further understand the evolutionary nature of a BI system, following section explains and discusses the evolving BI development methodologies.

2.3.1 Evolutionary release methodology of BI systems

According to Maddison (1983, cited in Avison & Fitzgerald 1999, p. 418), systems-development methodology is a "*recommended collection of philosophies, phases, procedures, rules, techniques, tools, documentation, management and training for developers of information systems*". There are various methodologies available, and they range from general methods, such as structured systems analysis and design methods, to proprietary methods. Since the 1960s, there have been fundamental changes in methodology concepts. Fitzgerald (1997) argues that the development of theories of system methodologies is currently being led by professional practice. This has usually been the case as it takes time for academics to scrutinise and generalise the practice into theory, as evidenced by BI system literature (O'Donnell, Arnott & Gibson 2002).

In line with the evolutionary notion, the implementation of a BI system is not a one-off IT-driven project, but an enterprise-wide *evolving* environment that is continually improved and enhanced in response to feedback from the business community (Moss & Atre 2003). Thus, the traditional development practices of operational systems are inadequate and inappropriate for BI systems.

In the past, transactional systems were never designed with integration in mind. Each system was designed to serve a single purpose for a functional set of people in a ‘single-swim-lane’ practice (Moss & Atre 2003). Traditional methodologies typically start with a functional business need, then concentrate on design and development, and finally end in maintenance. As a result, conventional ‘waterfall’ methodologies are considered sufficient for such static stand-alone systems. However, these traditional methodologies do not include strategic, enterprise-level planning, or cross-functional analysis; nor do they embrace the concept of application releases (Moss & Atre 2003).

Unlike a static stand-alone system, a dynamic, evolutionary and integrated BI system cannot be built in one ‘big bang’ (Moss & Atre 2003; Arnott 2004; Arnott & Pervan 2005). Based on the evolutionary notion, Moss and Atre (2003) propose a so-called ‘BI application release methodology’. According to them, BI applications must be developed and rolled out in an incremental delivery approach, in so-called *iterative* releases, and each deployment is likely to trigger new requirements for the next release, as illustrated Figure 2.9. Based on the features in this diagram, a summary of the differences in developmental methodology between a BI system and a conventional operational system is as follows (Moss & Atre 2003):

- BI applications are mostly triggered and driven by *business opportunities*.
- BI applications implement a *cross-functional* decision-support strategy rather than departmental decision-support silos.

- BI requirements are mostly *strategic information* requirements rather than operational functional requirements.
- Analysis of BI projects emphasises *business analysis* rather than system analysis, and analysis is the most important activity when developing a BI decision support environment.
- Ongoing BI application release evaluations promote *iterative* development and the *software release* concept rather than a single big development.

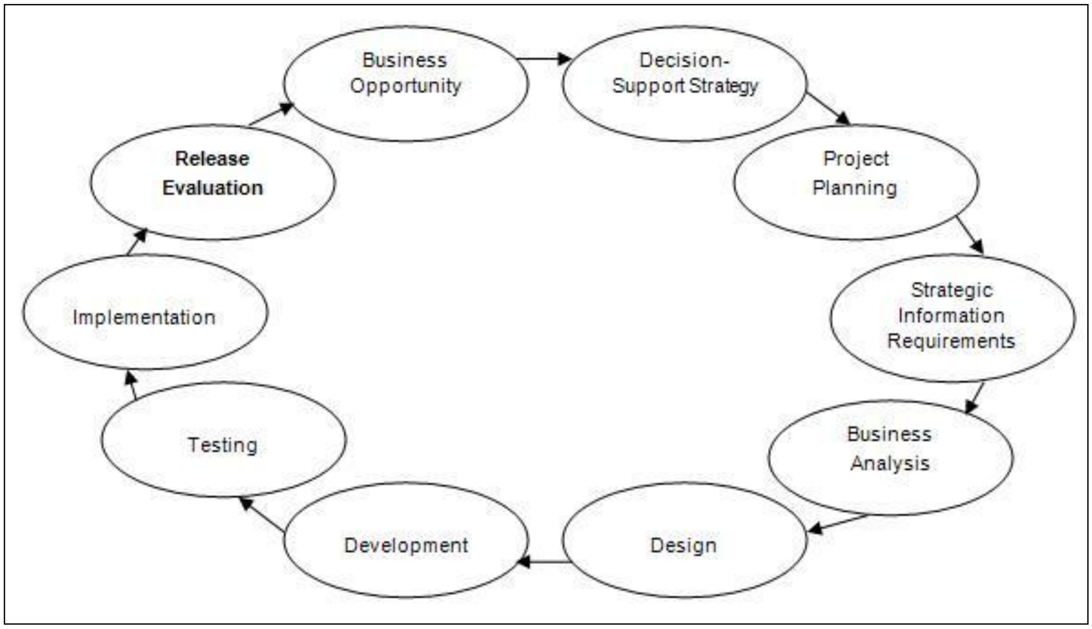

Figure 2.9: The BI application evolutionary release methodology

Source: *Adopted from Moss and Atre (2003)*

Moss and Atre (2003) outline an engineering roadmap, known as a *BI roadmap*, comprising agile and adaptive steps for simultaneous multiple sub-projects - where each can go through several of its own iterations or releases. Figure 2.10 illustrates the BI roadmap which consists of six major stages common to a typical BI project.

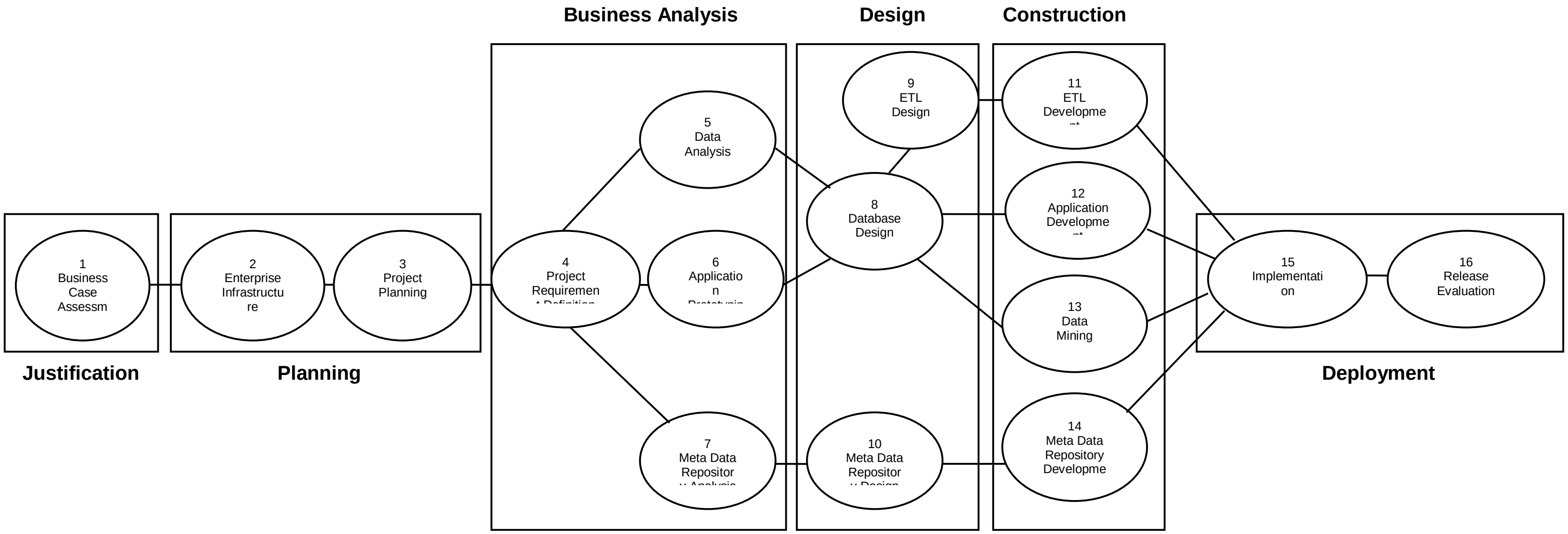

Figure 2.10: The BI roadmap with step dependencies

Source: *Adopted from Moss and Atre (2003)*

The interactions and steps in parallel development tracks. Moss and Atre (2003) state that most BI project teams are likely to implement the steps in parallel order. However, owing to the natural order of progression from one major stage to another, certain dependencies do exist between some of the development steps. As depicted in Figure 2.10, steps stacked on top of each other can be executed simultaneously, whereas steps that appear to the left or right of each other are performed relatively linearly.

A typical BI project may comprise at least three development tracks running in parallel following the definition of requirements but before actual implementation (Moss & Atre 2003). The three tracks are the ETL track, the application track, and the meta data repository track, and they are the major sub-projects in which each may have its own team members and set of activities. Figure 2.11 shows the interaction of the three tracks that eventually contribute to the overall BI project objectives.

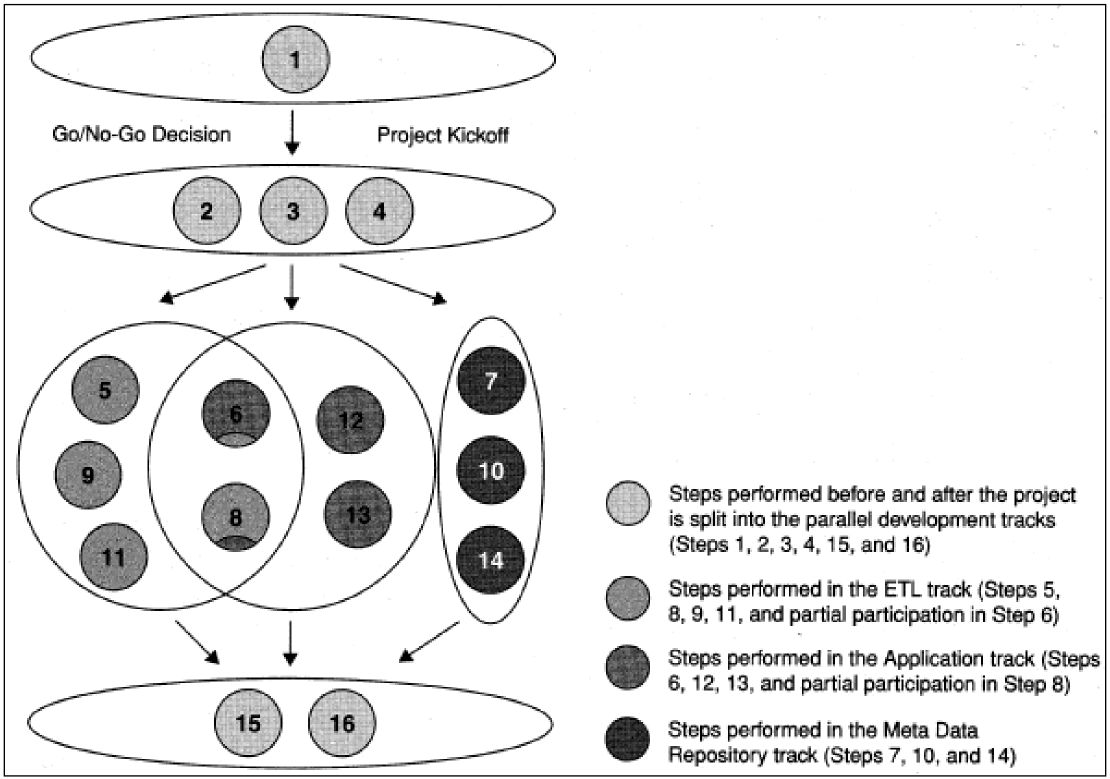
1
Go/No-Go Decision
Project Kickoff
2
3
4
5
9
11
6
8
12
13
7
10
14
15
16
Steps performed before and after the project is split into the parallel development tracks (Steps 1, 2, 3, 4, 15, and 16)
Steps performed in the ETL track (Steps 5, 8, 9, 11, and partial participation in Step 6)
Steps performed in the Application track (Steps 6, 12, 13, and partial participation in Step 8)
Steps performed in the Meta Data Repository track (Steps 7, 10, and 14)

Figure 2.11: The interactions and steps in parallel development tracks

Source: *Adopted from Moss and Atre (2003)*

Project-specific versus cross-organisational steps. During the implementation process, Moss and Atre (2003) further noted that substantial developmental steps must be performed from a cross-organisational perspective. Hence the project activities undertake a cross-functional dimension, and the participants of those activities should include domain experts from other lines of business who need to ratify and validate the policies, strategies, business rules and standards of the BI project. Table 2.5 lists the cross-organisational steps.

Table 2.5: Project-specific vs cross-organisational steps

Source: *Adopted from Moss and Atre (2003)*

Development step	Cross-organisational	Project-specific
1. Business case assessment	✓	
2. Enterprise infrastructure evaluation	✓	
3. Project planning		✓
4. Project requirement definition		✓
5. Data analysis	✓	
6. Application prototyping		✓
7. Meta data repository analysis	✓	
8. Database design	✓	
9. ETL design	✓	
10. Meta data repository design	✓	
11. ETL development	✓	
12. Application development		✓
13. Data mining	✓	
14. Meta data repository development	✓	
15. Implementation		✓
16. Release evaluation	✓	

In addition to these, Dresner *et al*. (2002) of Gartner Research identify nine significant steps in the lifecycle of a BI deployment (see Figure 2.12). They derive this methodology and lifecycle model from the best practices established by enterprises that have successfully implemented BI. Essentially, the model embraces the ideas and concepts associated with concurrent engineering and 'iterative' development methodologies. In contrast to the conventional structured system development methodology, this model replaces those traditional barriers with a more-fluid cycle in which all parties are simultaneously involved in the various steps required to achieve rapid 'time to action' (Dresner *et al*. 2002).

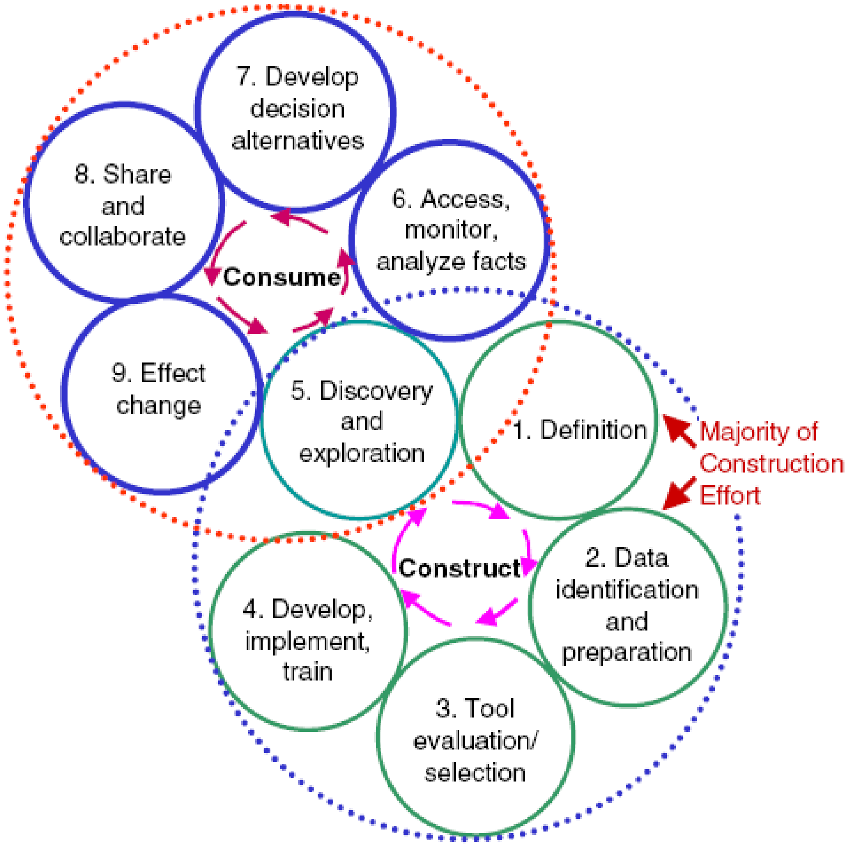
7. Develop decision alternatives
8. Share and collaborate
6. Access, monitor, analyze facts
Consume
9. Effect change
5. Discovery and exploration
1. Definition
Majority of Construction Effort
Construct
2. Data identification and preparation
4. Develop, implement, train
3. Tool evaluation/ selection

Figure 2.12: BI methodology and life cycle

Source: *Adopted from Dresner et al. (2002)*

As depicted in Figure 2.12, the BI lifecycle model is divided into two distinct, but intersecting, cycles: *construction* and *consumption* (Dresner *et al*. 2002). These cycles often operate at different rates and are iterative in nature. Resources and inputs from the BI competency team, the IT personnel, and the user cohort are required at different points in the cycle and with varying degrees of involvement. Dresner *et al*. (2002) recommends that at the inception of any new BI initiative it is vital to begin with a definition of requirements (Step 1) and to then follow the steps in order. Nevertheless, this model is only a general guide. Enterprises should adapt it to their specific needs and organisation.

In short, the use of a BI system development methodology offers a useful tool for understanding and promoting the sequence of steps for successfully implementing BI systems. The methodology can also serve as a guide in the application and alignment of resources and funding.