

# An Effective SDLC: Balancing Delivery and Control

**Dave Friesen, CMA, CISA**

Director of Application Services  
Epiq Systems Class Action & Claims Solutions

# Agenda

**The SDLC**

**Methods**

**“Universal” Activities**

**Wrap-up**

The [obvious] case for *process*

**Building great *business systems* is a challenge. . .**

**lack functionality**  
**are not flexible/extensible**  
**are not user friendly**  
**lack controls**  
**are unreliable**  
**underperform**  
**are difficult to maintain**

# The ***SDLC***

**A *[reasonable, consistent]*  
system delivery process**

**Useful  
High-quality  
Economical (build, COO)  
Efficient  
Secure**

# **Predictive** models: “Traditional”

**Stable requirements**

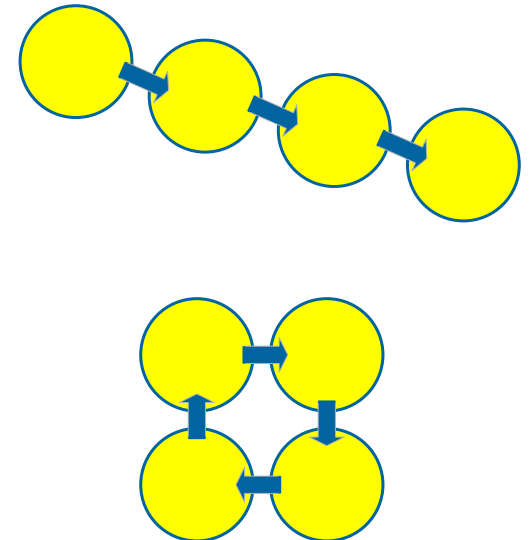
**Complex, high-risk systems**

**Formal documentation**

**Stable technology base**

**Structured; ~large teams**

**Can be iterative**



# ***Adaptive*** models: “Agile”

**Evolving requirements**

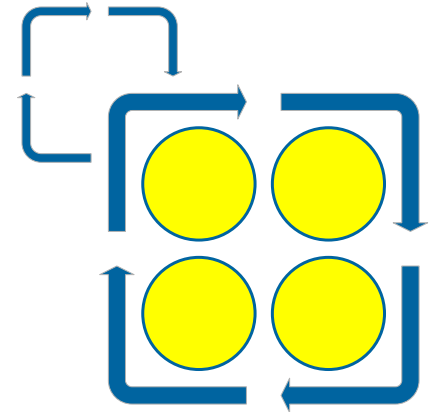
**Document ‘lite’**

**Iterative, short ‘bursts’**

**Code as the measure**

**Small teams; low formality**

**Communication**



# What are the *functional requirements*?

**The “what”;  
features/functionality**

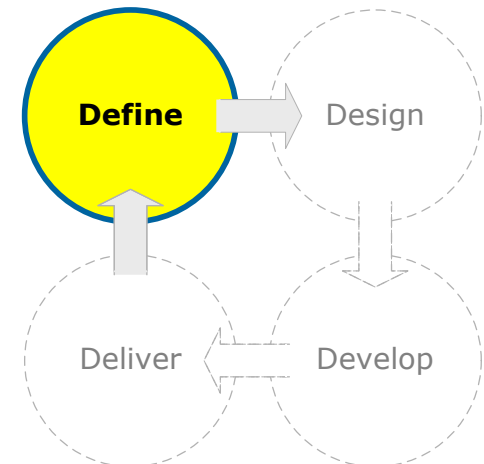
**Priorities**

***Specs***

***Use cases; release plan***

***User stories; backlogs***

**QA!**



Are *interfaces* considered?

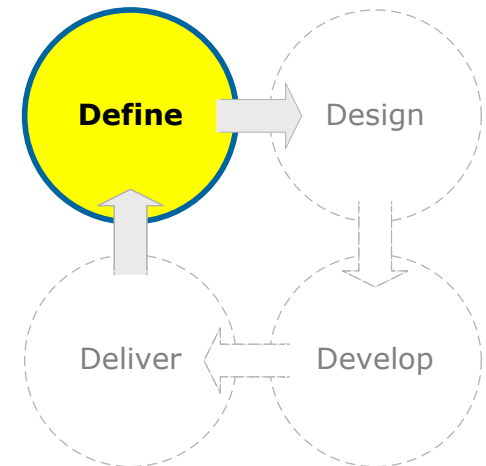
**Guaranteed delivery**

**Non-repudiation**

**Security**

**Reporting**

**Complete, Accurate, Valid**



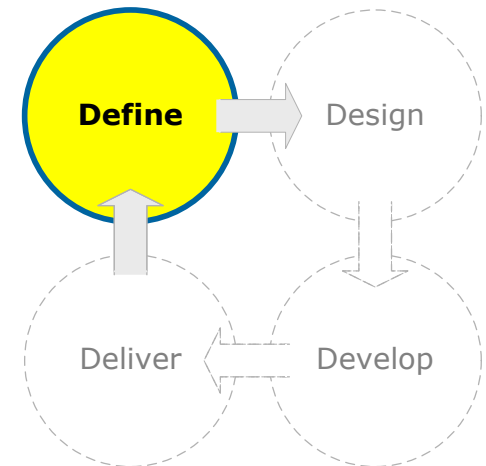
Are *application controls* considered?

**Inputs**

**Processes**

**Outputs**

*Vendor-provided controls?*



What are the *operational requirements*?

**Security**

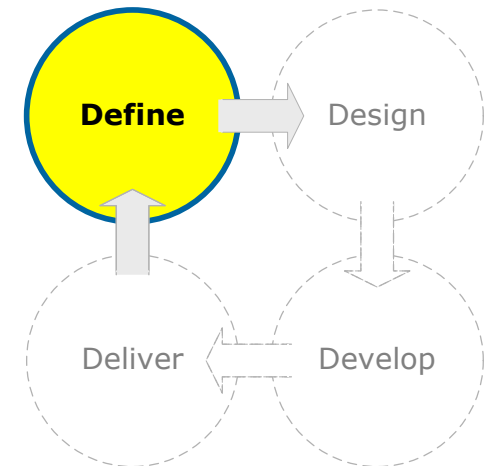
**Performance/reliability**

**Usability**

**Efficiency**

**Maintainability**

**Portability**



Is *security* considered?

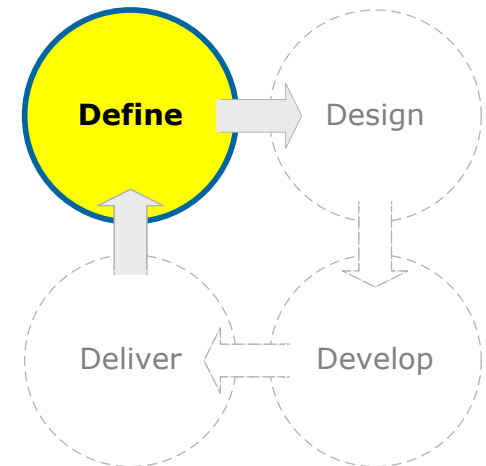
**Confidentiality**

**Integrity**

***Access methods***

***Audit trails***

***Architecture***



# A note about *project management*

**Manage scope &  
expectations**

**Communicate**

**Manage exceptions & risk**

**Manage cost**

# *Designing* the solution

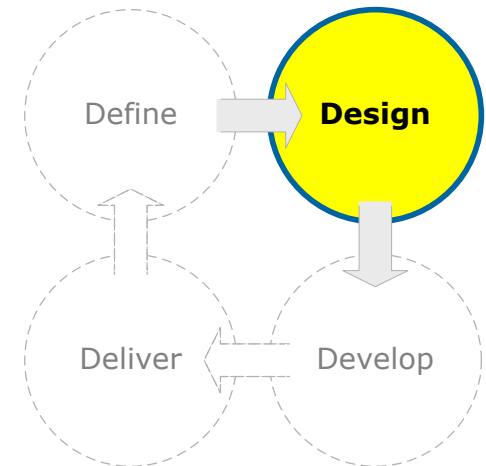
## The “how”

***Formal*** (structured) ***methods***

***Informal methods***

**QA**

**Security**



# ***Building*** the solution

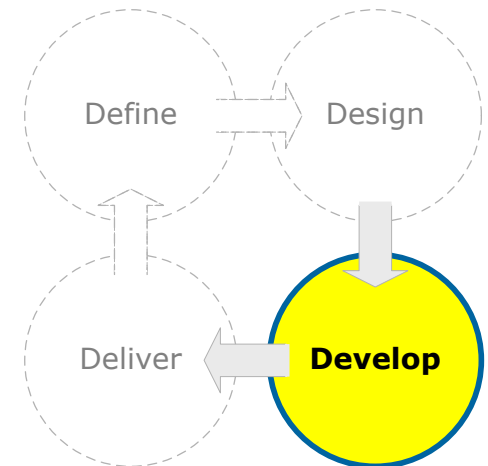
**Coding/configuring**

**Integrating**

**Writing tests**

**Re-working**

**Security**

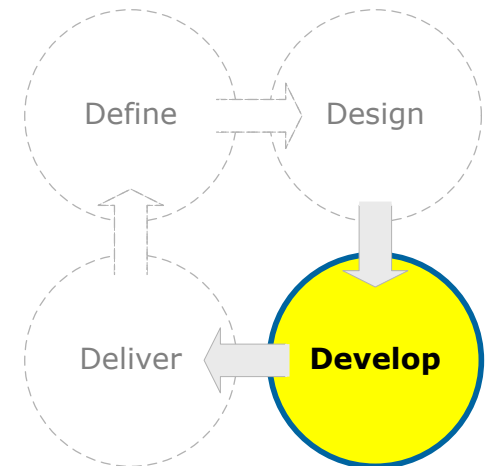


# *Environments? Source code control?*

**Dev/Test/QA/Prod**

**SOD**

***Formal tools, policies***



What are the **code** and tool **standards**?

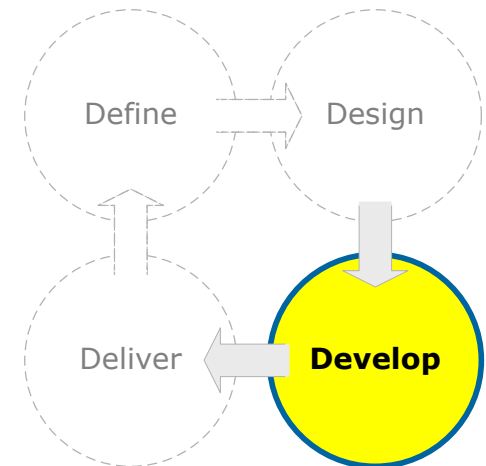
**Cost**

**Quality**

**Communication**

**Maintenance**

**Turnover**



# To what level is *QA/testing* performed?

**Unit**

**Integration**

**System**

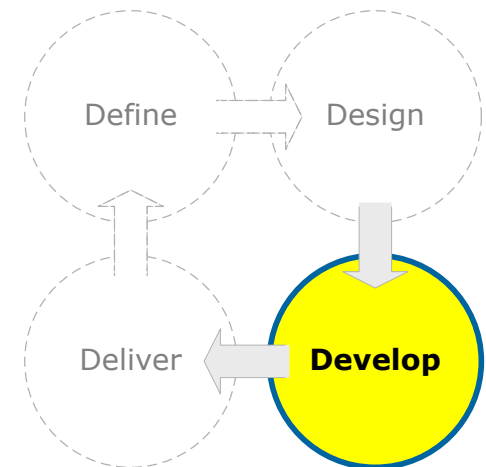
**Interfaces**

**Security**

**Stress+Volume; Performance**

**Usability**

**Regression**



Is *user acceptance testing* performed?

**Plan?**

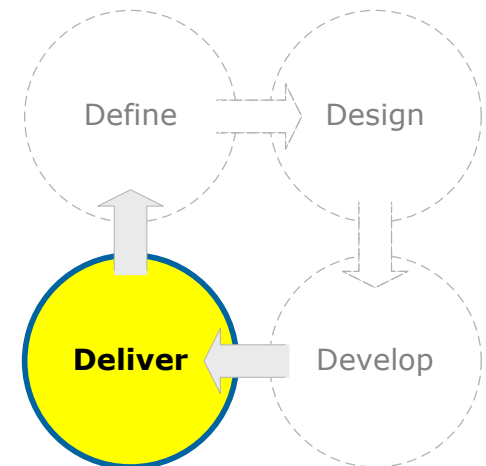
**Functional requirements**

**Operational requirements**

**Audit and controls**

**Exception handling**

**Sign-off**



# Other *key pre-release questions?*

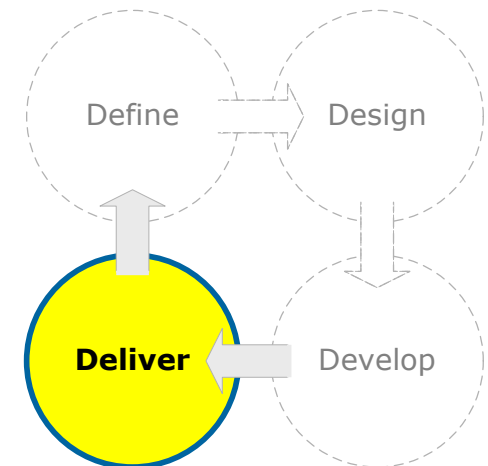
**Users trained?**

**Support in place?**

**Operational environment  
ready?**

**Security reviewed?**

**Data conversions complete?**



***Release*** (“Live”)

**Review; Iterate**

**Maintenance**

**Change management**

**Support**

# SDLC *considerations*

**Stability of requirements**

**Documentation needs**

**Culture**

**Team experience**

**Team size, location**

**Project size**

**Nature of technology**

# SDLC *must-haves*

**Quality**

**Data integrity**

**Business/IT partnership**

**Continuous improvement**

Questions/Discussion?

**Dave Friesen, CMA, CISA**

dave@davefriesen.com