Basic questions that apply to everything

Why – How – What - Who – Where – When (start with why)

Time based aspects

- Is this the first time (new) or a repeat (existing)?
- How likely is reuse: how soon, how often, by whom?

Structure vs Behaviour

(Think about both) Scenarios!

Performance Influence Matrix

Think about factors relevant to performance.

	#Users	**Transactions** (Handled by a package)	**Data-Flow** (between packages)
Frequency (peaks, etc)			
Volumes (totals, etc)			

1. Throughput – Transactions per unit of time.
2. Performance – Elapsed time per transaction.
3. Latency – Wait time for a response.
4. Capacity – Number of users / entities the system can support for a given configuration at a fixed level of performance.
5. Scalability – Ability to increase capacity.
6. Reliability – Length of time a system can operate without failure.
7. Response Time – Performance as perceived by a human.

Security Matrix

Factors relevant to security

1. Protection of data, both at rest and in-flight
2. Protection of resources (misuse of services / functionality)
3. Protection of systems (DOS, Out-of-band attacks, etc)
4. Boundary defence vs in-depth (e.g. encryption)
5. Authentication
6. Authorization
7. How will Authentication & Authorization be enforced, be managed?
8. Identity Management

Logical Layers (not including hardware)

1. User Interface
2. Application Layer (Service Layer or Controller Layer)
3. Domain Layer (Business Layer, Business logic Layer or Model Layer)
4. Infrastructure Layer (data access, logging, network access, security, email, file system, and so on)

Lifecycle

1. Design, Prototyping
2. Development / Refactoring
3. Testing (suitability, integration, performance, …)
4. Deployment / Use

Management Matrix: Who owns and controls what?

The point here is to take the concepts of Stewardship and Custodianship and apply it to the different parts of the stack.

	Ownership	Control
The process (that the system implements)		
The hardware		
Platform (OS)		
The system(s) (software / services)		
The data		

Business / Project Management

- What are the Critical Success Factors?
- Is there a common terminology across all involved parties?

Information Architecture

1. Organization / Structure
2. Labelling
3. Navigation, Browsing
4. Searching

Information Lifecycle

- Create, Read, Update, Delete

Transactions

- Atomic, Consistent, Isolated, Durable

Architecture -> Capabilities -> Features

- **Execution qualities**: such as security and usability, are observable at run time.
- **Evolution qualities**: such as testability, maintainability, extensibility and scalability, are embodied in the static structure of the software system.

Design Thinking

- Lenses: Desirability (people), Feasibility (technical), Viability (business)
- Steps: Learn from people, Find patterns, Generate ideas, Make tangible & prototype.

Some System Quality Attributes

1. **Accuracy**: Indicates proximity to the true value (how close are you).
2. **Precision**: The repeat-ability or reproduce-ability of the measurement (consistency within certain bounds).
3. **Modifiability**: Requirements about the effort required to make changes in the software. Often, the measurement is personnel effort (person- months).
4. **Portability**: The effort required to move the software to a different target platform. The measurement is most commonly person-months or % of modules that need changing.
5. **Robustness**: the quality of being able to withstand stresses, pressures, or changes in procedure or circumstance.
6. **Performance**
    A. See "Performance Influence Matrix"
    B. Expected transaction & response times (average, worst case)
        - response times
        - transaction rates
        - throughput
7. **Availability**
    A. Service Level Agreements
    B. Percentage of time available and/or hours of availability
    C. Expected recovery time
8. **Compatibility**
    A. Backwards compatibility.
    B. Dependencies – do they change between versions.
9. **Extensibility**: New capabilities can be added to the software without major changes to the underlying architecture.
10. **Modularity**: Well defined components, separation of concerns.
11. **Maintainability**: the ease with which a software product can be modified in order to:
    A. Correct defects
    B. Meet new requirements
    C. Make future maintenance easier
    D. Cope with a changed environment

Operating constraints

- System resources (storage, memory)
- People (hours of business, location)
- Software (dependencies, minimum requirements)

Deployment

- Physical location
- Political location (e.g. hosted internally or externally)
- Ease of access (maintenance) vs. Security
- Dependencies
- Communication between locations/servers/etc (how, management matrix, etc)

Levels of Granularity

[Macro <] Systems / Components / Classes [> Micro]

Some possible Views & Perspectives

1. Logical (typically coarse grained view)
2. Functional (fine grained logical view)
3. Deployment / Physical
4. Business Process (and specific roles within that)
5. Specific scenarios (both business and technical)
6. Application (Class / Module / Package / Component / Service)
7. Run-time (Concurrency / processes / threading)
8. The User.
9. Various stakeholders
10. Data
11. Security / Attack Surface
12. Public / private, internal / external

Intellectual Property

Asset types: brands, contracts, licences, design, copyright, databases.

Other things to consider

1. Alignment with current and future technology stacks, industry trends
2. User Interfaces
3. Backend Interfaces / integration
4. Context
5. Responsibilities (both within the systems, and teams delivering them)
6. Design Patterns
7. Platform(s)
8. Appropriate Granularity
9. Supportability
10. Maintainability (evolvement – extension – refactoring)
11. Dependencies (on other parties, systems, services, standards)
12. Interoperability adherence to standards, or: is achieved when the coherent, electronic exchange of information and services between systems takes place.
13. Portability
14. Resilience / Robustness: the quality of being able to withstand stresses, pressures, or changes in procedure or circumstance.
15. Resource constraints (processor speed, memory, disk space, network bandwidth etc.)
16. Scalability (horizontal, vertical / out, up)
17. Security
18. Usability by target user community