

ENGINEERING PROGRESS MANUAL





CONTENTS

1. Purpose, structure and goal

1.1 Purpose of the area

1.2 Roles and responsibilities

1.3 Philosophy

1. Rules analysis and breakdown

2.1 Rules immersion

2.2 Scoring breakdown and strategy

2.3 Defining the robot's scope

1. Conceptual design

3.1 Mechanism research

3.2 Brainstorming

3.3 Low-fidelity prototyping

3.4 Design evaluation and freeze

1. Robot Requirements Document (RRD).

4.1 Writing the RRD

4.2 RRD review and validation

4.3 Handoff to CAD modeling

1. Documentation

5.1 Document control

1. Phase closeout





PURPOSE, STRUCTURE AND GOAL

This document defines the process the team follows during the first phase of the season, from kickoff to the Robot Requirements Document (RRD).

The goal is to make sure the robot design:

- Responds directly to a competitive strategy
- Is based on analysis and data
- Is validated before moving to CAD

The process is sequential and logical: first understand the game, then define how to win, then design solutions, validate them and finally document them.

1.1 PURPOSE OF THE AREA

The engineering area is responsible for turning the game challenge into a functional, competitive and viable robotic system.

This means:

- Interpreting the rules correctly
- Translating strategic needs into technical requirements
- Designing efficient solutions
- Validating decisions through prototypes

The goal is not to build fast, but to build with certainty.

1.2 ROLES AND RESPONSIBILITIES

To avoid disorganization, every role has clear duties:

- Engineering lead: oversees the technical process and validates key decisions.
- Captain / strategy lead: defines how the game is won and prioritizes tasks.
- Design lead: coordinates the development of mechanisms and concepts.
- Sub-teams: research solutions for specific tasks.
- Prototyping team: builds and tests mechanisms.
- Documentation lead: records the whole process and keeps it organized.

1.3 PHILOSOPHY

The process follows key principles that apply at all times:

- Strategy before design: we never design without understanding the game.
- Data before opinions: every decision needs a justification.
- Fast iteration: it is better to fail in a prototype than at competition.
- Simplicity: reliability always beats complexity.





RULES ANALYSIS AND BREAKDOWN

The rules analysis phase builds a complete, accurate and shared understanding of the season challenge and sets the strategic foundation the robot will be designed on.

This stage feeds every later decision. An incorrect or incomplete analysis leads to weak strategies and, as a result, inefficient designs.

The goal is not only to understand what the rules allow or forbid, but to find how to maximize performance within their limits.

Phase approach

The analysis takes both a technical and a strategic view, considering:

- Which actions create the most value in the game
- Which constraints limit the robot design
- Which opportunities can be exploited competitively
- How a match actually plays out

This phase keeps understanding the problem separate from generating solutions. No mechanisms or design ideas are proposed at this point.

Analysis components

The process has two main sub-stages:

2.1 Rules immersion

In this sub-stage the team does a deep, structured reading of the official game manual to identify:

- Technical and operational constraints
- Relevant penalties
- Field elements and how they interact
- Overall game flow

The result is a clear, documented interpretation shared by the whole team.



SCORING BREAKDOWN AND STRATEGY

This is where we answer the most important question: how is the game won?
The analysis must be quantitative, not based on intuition, and must include the data behind it.

Prioritization matrix

This tool lets us evaluate every game action objectively:

Tarea del juego	Valor de puntos (VP)	Frecuencia esperada (FE)	Riesgo/Dificultad (R) 1-5	prioridad (d) $d=(VP*FE)/R$	Notas
Ejemplo: colocar una pieza	5	6 ciclos durante una partida	2		debe ser un robot rápido

Definitions:

- PV: point value
- EF: expected frequency per match
- R: difficulty or risk
- d: priority level

Additional analysis criteria

Beyond the prioritization matrix, the analysis must include operational variables that show how viable each task really is in a competition setting. These variables move us from a theoretical analysis to an estimate much closer to on-field performance.

Cycle time

Cycle time is the time needed to complete an action from start to finish, including every intermediate step (acquiring, positioning, executing and resetting).

Analyzing it lets us:

- Estimate the maximum number of times a task can be repeated per match
- Compare efficiency across actions
- Find bottlenecks in execution

We recommend breaking each task into subtasks and estimating partial times for a more accurate evaluation.





Expected reliability

Reliability is the probability that an action is completed successfully under real competition conditions.

This factor must consider:

- Variability in execution (human or system errors)
- Interaction with other robots
- Field conditions
- Complexity of the required mechanism

A high-value task with low reliability can be less effective than a simpler but highly consistent one. That is why reliability is weighted as a critical decision factor.

Operational dependencies

Dependencies describe the sequential or conditional relationship between robot actions.

For example:

- You cannot score without first acquiring a game piece
- Some actions require a specific position before they can run
- Some tasks can block or limit others from running at the same time

Dependency analysis lets us:

- Understand the real complexity of each task
- Find the most efficient execution paths
- Avoid loading the design with unnecessary chained processes

Interaction with the field and other robots

We also consider how each task is affected by external factors:

- Allied or opposing robots nearby
- Possible defensive scenarios
- Congestion in key field zones

This analysis anticipates real game conditions and leads to more robust strategies.





Analysis output

At the end of this stage, the team has turned the rules and scoring analysis into a clear, actionable strategic foundation.

Specifically, it produces:

Prioritized task list

An ordered list of every possible game action, ranked by strategic impact, considering both point value and operational viability.

The list shows not only which actions are worth the most, but which are the most efficient and consistent in a real competition.

Defining the robot's strategic core

From the prioritized list, the team selects the **three main tasks** that will define the robot's behavior during the season.

This strategic core must:

- Maximize point generation
- Be achievable within the available development time
- Balance complexity and reliability

These tasks are the foundation for the entire robot design.

Assigning research sub-teams

Once the strategic core is defined, sub-teams are assigned to research solutions for each priority task.

Each sub-team will:

- Analyze possible mechanisms
- Evaluate implementation alternatives
- Prepare proposals for the conceptual design phase

This lets the team move forward on several fronts in parallel, saving development time and building specialization within the team.

Success criteria

This stage is complete when:

- There is a clear, justified prioritization shared by the whole team
- A concrete strategic approach for the robot has been defined
- Specific responsibilities have been assigned for the next phase





DEFINING THE ROBOT'S SCOPE

Once the rules analysis and strategic prioritization are done, the robot's scope must be stated explicitly.

This section defines clearly what the robot will and will not do, keeping the design focused, viable and aligned with the team's strategic goals.

Scope definition works as a control that prevents feature overload (over-engineering) and makes later design decisions easier.

Approach to defining scope

The robot's scope is based directly on:

- The prioritized list of game tasks
- The defined strategic core (top 3 tasks)
- The team's technical capabilities and resources
- The available development time

Not every game task has to be built. A competitive robot does a few things consistently and efficiently; it does not try to do everything.

Included features

The team sets a clear list of the functions the robot will perform at competition.

These functions must:

- Come directly from the defined strategy
- Be technically viable
- Have a significant impact on scoring

Example:

- Pick up game pieces from the floor
- Carry game pieces securely
- Score in the main goal
- Run the endgame action

Every selected function must be justified by its strategic value.

Excluded features

The team must also define which functions will not be built, even if the game allows them.

This helps to:

- Avoid drifting during design
- Reduce unnecessary complexity
- Keep the team focused





Example:

- No scoring in the secondary goal
- No active defensive capability
- Low-frequency or high-risk functions are not prioritized

These decisions are documented along with their justification.

Selection criteria

Including or excluding a function is based on objective criteria such as:

- Impact on total score
- Required cycle time
- Expected reliability
- Implementation complexity
- Available resources

Priority goes to the functions that make the robot most efficient in real competition conditions.

Alignment with design

The scope defined here is the mandatory reference for:

- Mechanism research (3.1)
- Brainstorming (3.2)
- Prototyping (3.3)
- Writing the RRD (4.1)

Any proposal that does not fit the defined scope is discarded or re-evaluated.

Deliverables

At the end of this section, the team has:

- **List of included features**
- **List of excluded features**
- **Technical and strategic justification for each decision**

Success criteria

This stage is complete when:

- The robot's scope is completely clear
- There is no ambiguity about what will be built
- The team agrees on the design priorities

Strategic importance

A well-defined scope focuses the team's effort on what actually creates value, avoids spreading resources thin and lowers the risk of failures in later stages.





CONCEPTUAL DESIGN

3.1 MECHANISM RESEARCH

Before any design work, the team runs a structured research stage so technical decisions rest on existing knowledge and proven evidence.

The goal of this phase is to identify, analyze and evaluate possible mechanical solutions for each priority task, making sure proposals are not only functional but also feasible with the team's technical skills and materials.

This stage also explicitly accounts for the resources available in the workshop, such as reusable components, tools, the team's previous experience and manufacturing limits. Designing without these factors in mind greatly increases implementation risk.

Research approach

Research focuses on understanding how and why existing mechanisms work, rather than collecting examples at a surface level.

Each assigned sub-team approaches its research with these goals:

- Identify proven solutions for similar tasks
- Understand the mechanical principles involved
- Evaluate whether those solutions apply to the current challenge
- Spot possible improvements or adaptations

Sources of information

For a complete analysis, we recommend consulting several technical sources:

- Mechanisms from past seasons
- FIRST robots (FRC/FTC) that solved similar problems, looking for recurring design patterns and approaches.
- Solutions from competitive teams
- High-performing robots, paying special attention to simplicity, robustness and efficiency.
- Engineering books and technical documentation
- Theory behind how mechanisms work (transmissions, four-bar linkages, pulleys, pneumatics, etc.).
- Whitepapers and technical resources
- Documents by other teams or experts that detail design decisions, tests and results.





Mechanism evaluation criteria

Every mechanism identified is analyzed critically against these criteria:

- Simplicity
- Number of parts, ease of assembly and overall complexity. Simple solutions that reduce failure points come first.
- Reliability
- Ability to operate consistently under real competition conditions, including wear, impacts and field variability.
- Ease of implementation
- Feasibility of building the mechanism considering:
 - Available tools
 - The team's technical level
 - Fabrication time
 - Material availability
- Fit with team resources
- Whether the mechanism can be built with parts already in the workshop or needs new purchases.
- Scalability and integration
- Ability to integrate with the robot's other subsystems without space, weight or control conflicts.

Expected technical analysis

The analysis of each mechanism goes beyond a surface description. For each option, document:

- Operating principle
- Main components
- Advantages and disadvantages
- Potential risks
- Build requirements
- Possible failure points
- Visual references (diagrams, images or real examples)

This level of detail supports informed decisions in later stages.

Common mistakes to avoid

- Copying designs without understanding how they work
- Ignoring the team's real limitations
- Choosing complex solutions without strategic justification
- Not documenting sources or conclusions
- Overlooking future integration with other systems





Deliverable

The output of this phase is a structured mechanism catalog, organized by each priority task.

The catalog includes:

- Several alternatives per task
- Detailed technical analysis of each option
- Comparison between solutions
- Preliminary recommendations

This document is the direct input for brainstorming and conceptual design, so later proposals rest on solid knowledge instead of assumptions.

Well-executed research at this stage greatly reduces design uncertainty and speeds up development, letting the team focus on optimizing solutions instead of discovering them from scratch.





BRAINSTORMING

Brainstorming generates a wide range of possible solutions for each priority task, encouraging the team's creativity without premature limits. At this stage the focus is not on the immediate quality of proposals, but on the variety and number of solutions, so the team explores different approaches before any filtering or technical evaluation.

Methodology

The session runs in a structured setting that encourages active participation from every team member.

Each participant first generates proposals individually, then shares and discusses them with the group.

We recommend structuring the session like this:

1. Individual idea generation
2. Each member develops concepts independently, avoiding outside influences that limit creativity.
3. Presenting proposals
4. Each participant presents her ideas to the team, explaining how each mechanism generally works.
5. Central record
6. Every idea is documented in a shared space (physical or digital) so no proposal gets lost.
7. Initial technical discussion (no filtering)
8. Questions are welcome to understand each proposal better, but no negative judgments or discarding at this stage.

Key rules

For this stage to work, the team follows these guidelines:

- Quantity over quality
- More ideas increase the chance of finding the best solutions.
- Avoid destructive criticism
- No proposal is discarded too early; the goal is an environment of trust and openness.
- Record every idea
- Even ideas that seem unworkable can inspire more refined solutions later.
- Everyone participates
- Each member contributes at least 2 to 3 concepts per mechanism.





BRAINSTORMING

Sketch format

Each proposal is presented as a sketch that clearly shows how it works.

Sketches include:

- **Visual representation**
 - Hand-drawn or digital, as long as it communicates the idea effectively.
- **Detailed functional description**
 - Explanation of the mechanism's operating principle, including key elements such as:
 - System type (e.g. four-bar linkage, rollers, gear transmission)
 - Motion flow
 - Interaction with the game piece
- **Technical notes**
 - Relevant details on approximate dimensions, pivot points, directions of motion or critical components.

Minimum quality criteria

Although this phase values quantity, each proposal needs a basic level of clarity so it can be evaluated later:

- Other team members can understand the mechanism
- There is a clear link between the proposed design and the task it solves
- The explanation is enough to prototype it later

Common mistakes to avoid

- Generating ideas unrelated to the priority tasks
- Repeating concepts without meaningful variations
- Holding back creativity out of fear of unconventional solutions
- Not documenting sketches properly
- Turning the session into an evaluation instead of idea generation

Deliverable

The output of this phase is a **complete, organized set of ideas** that captures every possible solution the team generated.

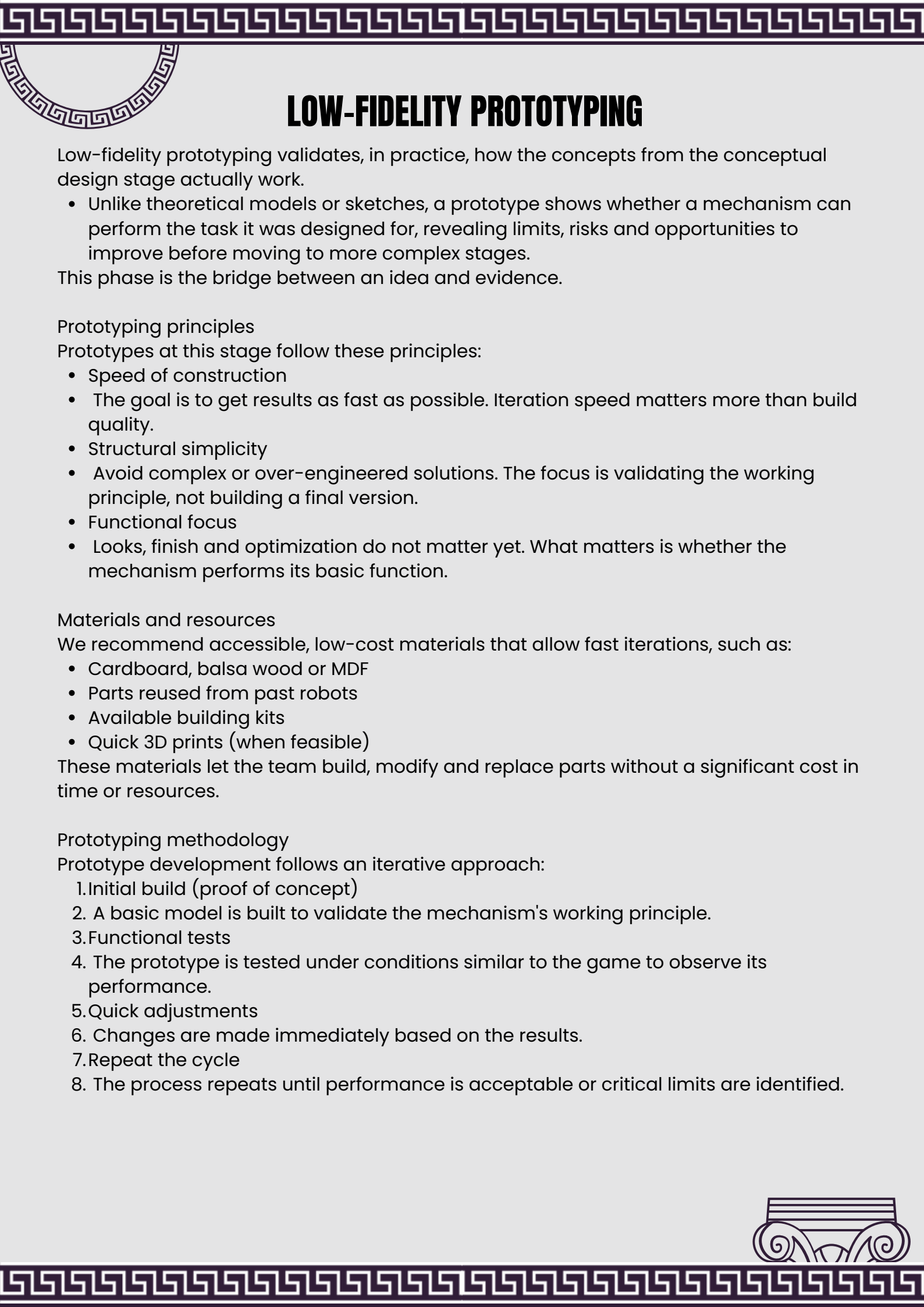
This set can be organized as:

- A physical wall of sketches
- A digital board (e.g. Notion, Miro, Figma)

The deliverable must make it possible to:

- See every proposal clearly
- Compare different approaches to the same task
- Serve as the basis for selection and prototyping in the next phase





LOW-FIDELITY PROTOTYPING

Low-fidelity prototyping validates, in practice, how the concepts from the conceptual design stage actually work.

- Unlike theoretical models or sketches, a prototype shows whether a mechanism can perform the task it was designed for, revealing limits, risks and opportunities to improve before moving to more complex stages.

This phase is the bridge between an idea and evidence.

Prototyping principles

Prototypes at this stage follow these principles:

- Speed of construction
- The goal is to get results as fast as possible. Iteration speed matters more than build quality.
- Structural simplicity
- Avoid complex or over-engineered solutions. The focus is validating the working principle, not building a final version.
- Functional focus
- Looks, finish and optimization do not matter yet. What matters is whether the mechanism performs its basic function.

Materials and resources

We recommend accessible, low-cost materials that allow fast iterations, such as:

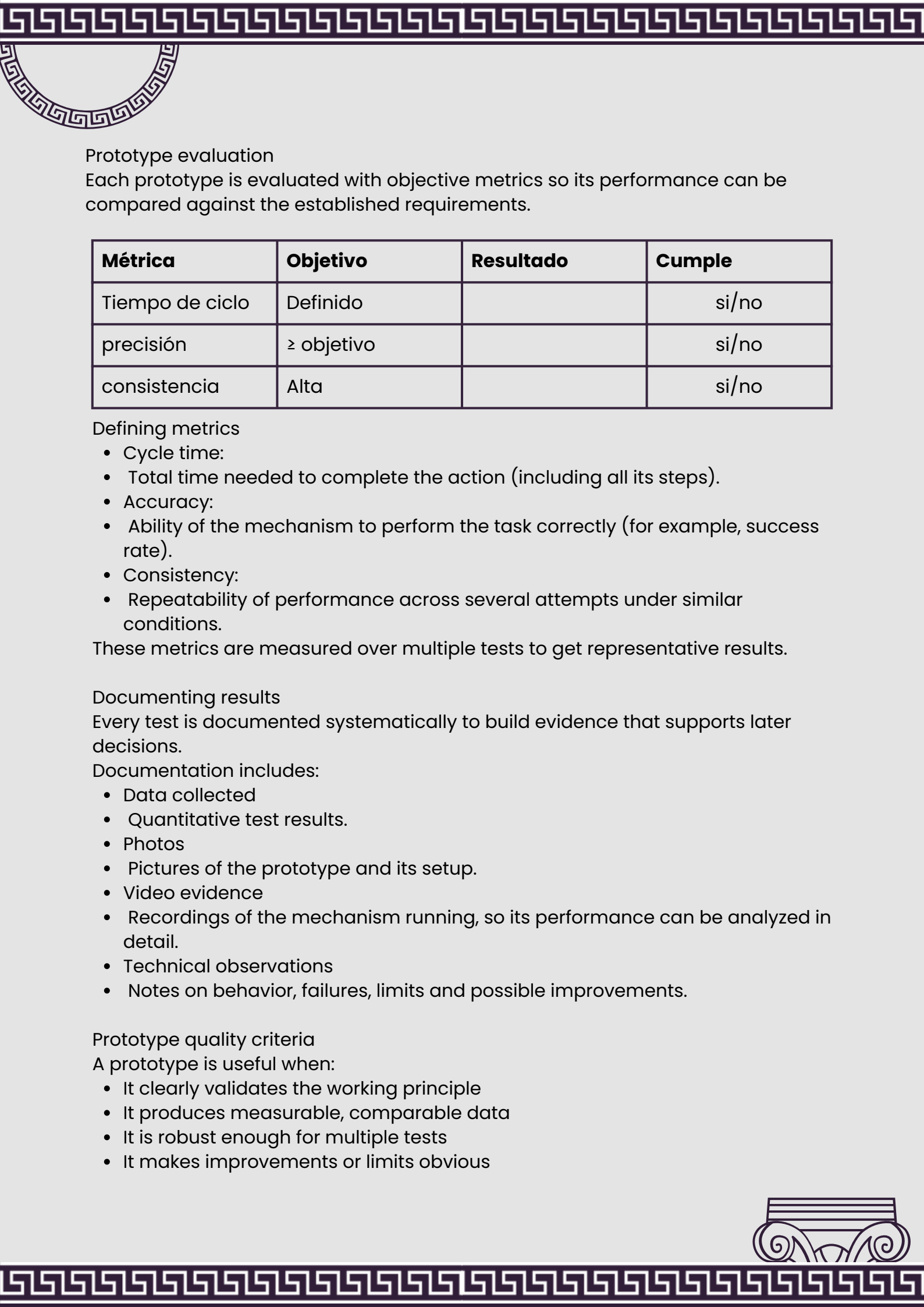
- Cardboard, balsa wood or MDF
- Parts reused from past robots
- Available building kits
- Quick 3D prints (when feasible)

These materials let the team build, modify and replace parts without a significant cost in time or resources.

Prototyping methodology

Prototype development follows an iterative approach:

1. Initial build (proof of concept)
2. A basic model is built to validate the mechanism's working principle.
3. Functional tests
4. The prototype is tested under conditions similar to the game to observe its performance.
5. Quick adjustments
6. Changes are made immediately based on the results.
7. Repeat the cycle
8. The process repeats until performance is acceptable or critical limits are identified.



Prototype evaluation

Each prototype is evaluated with objective metrics so its performance can be compared against the established requirements.

Métrica	Objetivo	Resultado	Cumple
Tiempo de ciclo	Definido		si/no
precisión	≥ objetivo		si/no
consistencia	Alta		si/no

Defining metrics

- Cycle time:
 - Total time needed to complete the action (including all its steps).
- Accuracy:
 - Ability of the mechanism to perform the task correctly (for example, success rate).
- Consistency:
 - Repeatability of performance across several attempts under similar conditions.

These metrics are measured over multiple tests to get representative results.

Documenting results

Every test is documented systematically to build evidence that supports later decisions.

Documentation includes:

- Data collected
- Quantitative test results.
- Photos
 - Pictures of the prototype and its setup.
- Video evidence
 - Recordings of the mechanism running, so its performance can be analyzed in detail.
- Technical observations
- Notes on behavior, failures, limits and possible improvements.

Prototype quality criteria

A prototype is useful when:

- It clearly validates the working principle
- It produces measurable, comparable data
- It is robust enough for multiple tests
- It makes improvements or limits obvious



Common mistakes to avoid

- Spending too much time perfecting the prototype
- Evaluating without clear metrics
- Running too few tests before deciding
- Not documenting results
- Mistaking a working prototype for a final design

DESIGN EVALUATION AND FREEZE

Once prototype testing is done, the team runs a formal evaluation to decide which concepts go into the final robot design.

The goal of this stage is to select the most viable solutions based on objective evidence and technical criteria, so the design that moves to CAD is validated, justified and aligned with the defined strategy.

The design freeze is a critical checkpoint in the engineering process: it sets the robot's definitive foundation and lowers the chance of rework later.

Evaluation process

The evaluation is structured and considers both the performance seen in prototypes and the feasibility of implementation at the system level.

1. Technical evaluation of each option

Each concept is analyzed individually using the results from prototyping.

The following aspects are considered:

- Meets the required function
- Measured performance (cycle time, accuracy, consistency)
- Stability across multiple tests
- Mechanical complexity
- Ease of control and operation
- Compatibility with other subsystems

This analysis relies on documented data, avoiding subjective decisions.

2. Comparison against performance metrics

Results are compared with the goals set earlier in the strategy phase.





We recommend evaluating each concept on:

- Performance: how well it completes the task
- Efficiency: performance relative to cycle time
- Reliability: consistency across repeated conditions
- Robustness: behavior under changing field conditions

This comparison shows which solutions come closest to the robot's requirements.

3. Go / No-Go decision

The final decision is made formal through a structured Go / No-Go process led by the technical team.

- GO (approved):
 - The concept meets the criteria and is viable to build.
- NO-GO (rejected or back to iteration):
 - The concept does not meet the requirements or carries significant risks.

Every decision is documented with its technical justification.

Success criteria

For a concept to be considered viable, it must reach at least:

- **75% of the defined target performance**

The team also evaluates:

- Performance consistency
- Room for improvement
- Impact on the full robot system

This criterion ensures the mechanism has a solid functional base before moving forward.

Possible outcomes

GO (design approved)

When a concept meets the criteria:

- **The mechanism's conceptual design is frozen**
- It is formally added to the Robot Requirements Document (RRD)
- It becomes the mandatory basis for CAD modeling

From this point on, any change must go through a formal approval process.





NO-GO (iterate or discard)

When a concept does not meet the criteria:

- The reasons for rejection are documented
- Areas for improvement are identified
- The team decides whether to iterate on the concept or discard it

When iterating, the process can go back to earlier phases to generate new solutions or refine existing ones.

Risk management

During this stage, the team identifies risks tied to approved concepts, such as:

- Fabrication complexity
- Space or weight limits
- Critical dependencies on other subsystems
- Possible failures under competition conditions

Spotting these risks early makes it possible to mitigate them before moving on.

Deliverables

At the end of this phase, the team has:

- List of selected concepts
- Mechanisms approved for the final robot design.
- Go / No-Go decision log
- Document with the evaluation and justification of each concept.
- Documented lessons learned
- Record of key findings, mistakes and recommendations for future iterations.

Phase closeout criteria

This stage is complete when:

- Every key subsystem has an approved concept
- Decisions are documented and justified
- It is completely clear what will be designed in CAD





ROBOT REQUIREMENTS DOCUMENT (RRD)

WRITING THE RRD

The Robot Requirements Document (RRD) is the central technical artifact that formally defines the robot's characteristics before CAD modeling starts.

Its purpose is to translate the game strategy and validated concepts into clear, measurable and actionable technical specifications, removing ambiguity and reducing uncertainty in later stages.

The RRD is not a descriptive document; it is a mandatory design guide.

RRD structure

The document is organized into clearly defined sections so it can be used as a direct reference for design, manufacturing and programming.

1. Functionality

This section states what the robot must do.

It is a structured, ranked list of tasks aligned with the strategy defined earlier.

Each function must:

- Be clear and specific
- Be tied to a measurable action
- Avoid ambiguity

Example:

- 1.1 Pick up a game piece from the floor in any orientation
- 1.2 Carry the game piece securely
- 1.3 Score in the elevated goal
- 1.4 Run the endgame action

2. Critical parameters

Defines how well each function must be performed.

These parameters must be quantifiable and serve as a direct reference for design validation.

At a minimum, include:

- Cycle time: maximum time allowed to complete a task
- Accuracy: minimum required success rate
- Speed: the robot's driving capability
- Reliability: expected execution consistency

Example:

- Cycle time: ≤ 10 s
- Accuracy: $\geq 80\%$
- Reliability: $\geq 90\%$ in repeated tests





3. Technology decisions

Defines the technical choices that shape the robot design.

These decisions are made in this phase to avoid inconsistencies during development.

Includes:

- Drivetrain type (tank, swerve, mecanum)
- Motor family (e.g. NEO, Kraken; avoid unnecessary mixing)
- Control system (roboRIO, Control Hub, etc.)
- Overall subsystem architecture

These decisions must be consistent with the strategy and the team's capabilities.

4. Constraints

Sets the limits the design must stay within.

Includes:

- Maximum dimensions (including extensions, if applicable)
- Target weight (not just the maximum allowed, but with a safety margin)
- Game manual limits
- Internal team limits (tools, budget, time)

The design must strictly meet these constraints.

5. Subsystem interfaces (recommended)

Defines how the robot's systems interact with each other.

Includes:

- Mechanical connections
- Game piece handoff between mechanisms
- Shared space requirements

This section reduces conflicts during modeling and assembly.

6. Design priorities (recommended)

Sets the order of importance between functions when they conflict.

Example:

1. Intake reliability
2. Cycle speed
3. Endgame capability

This makes decisions easier when space, weight or time are limited.





Core rule

- The RRD is a binding document.

The CAD design team:

- Cannot deviate from the RRD
- Must meet every specification in it

Any change requires:

- Documented technical justification
- Assessment of the impact on the system
- Formal approval by the technical team

Deliverables

At the end of this phase, the team has:

- A complete, structured RRD
- A clear, precise and unambiguous document
- Formal approval by the technical leads (engineering, design and strategy)

4.2 RRD REVIEW AND VALIDATION

Before release, the RRD goes through a formal review to ensure its quality and consistency.

Review process

The review involves:

- Engineering lead
- Design lead
- Strategy captain
- Technical mentor (if applicable)

Each participant confirms that:

- The document correctly reflects the game strategy
- The defined parameters are achievable
- There are no contradictions between sections
- Every function is technically viable

Validation criteria

The RRD is valid when it is:

- Complete (no key functions are missing)
- Clear (it cannot be read more than one way)
- Measurable (every parameter is quantifiable)
- Viable (it can be built with the available resources)





Core rule

- The RRD is a binding document.

The CAD design team:

- Cannot deviate from the RRD
- Must meet every specification in it

Any change requires:

- Documented technical justification
- Assessment of the impact on the system
- Formal approval by the technical team

Deliverables

At the end of this phase, the team has:

- A complete, structured RRD
- A clear, precise and unambiguous document
- Formal approval by the technical leads (engineering, design and strategy)

4.2 RRD REVIEW AND VALIDATION

Before release, the RRD goes through a formal review to ensure its quality and consistency.

Review process

The review involves:

- Engineering lead
- Design lead
- Strategy captain
- Technical mentor (if applicable)

Each participant confirms that:

- The document correctly reflects the game strategy
- The defined parameters are achievable
- There are no contradictions between sections
- Every function is technically viable

Validation criteria

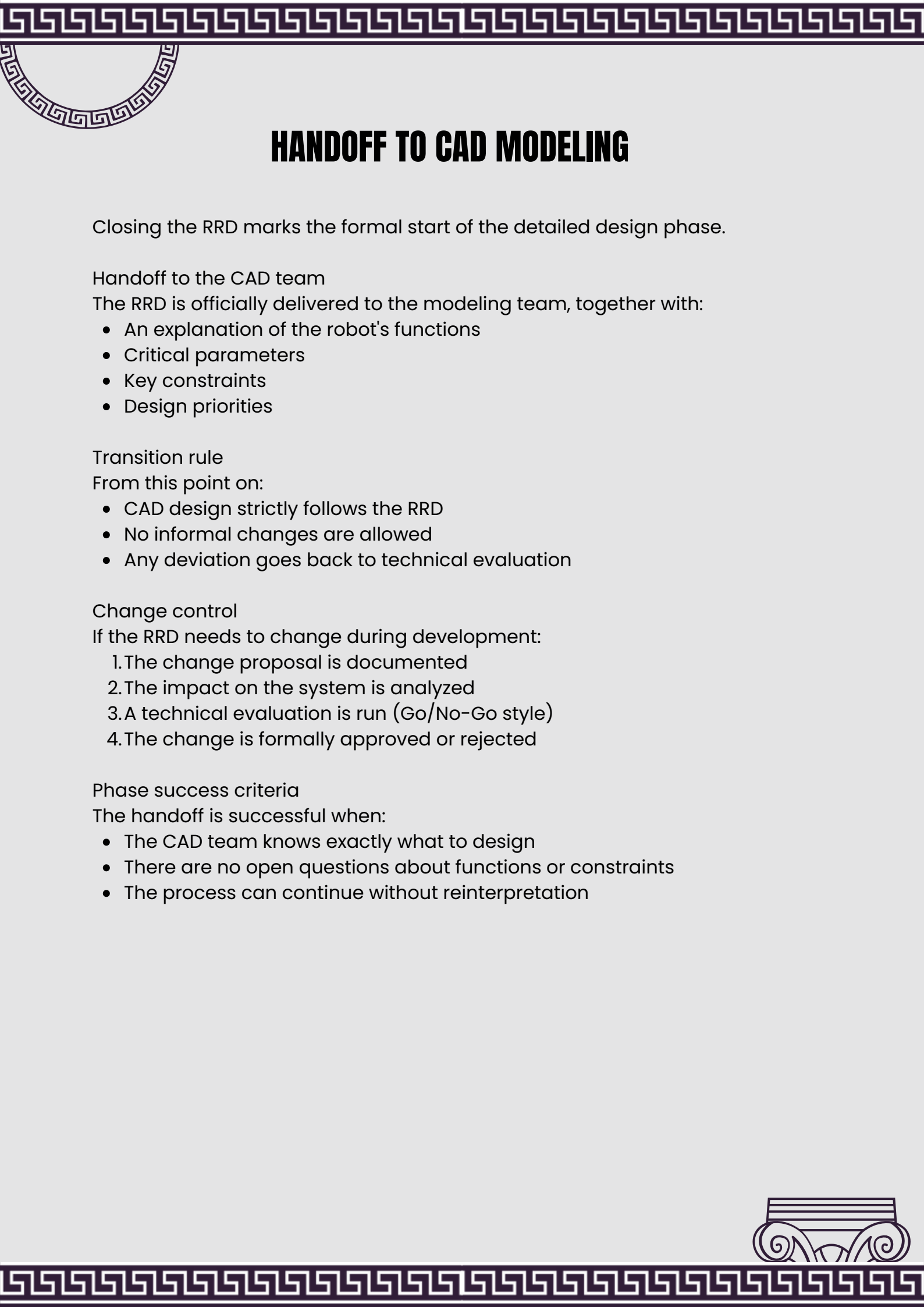
The RRD is valid when it is:

- Complete (no key functions are missing)
- Clear (it cannot be read more than one way)
- Measurable (every parameter is quantifiable)
- Viable (it can be built with the available resources)

Outcome

- Document approved for design use
- Review log with observations and corrections





HANDOFF TO CAD MODELING

Closing the RRD marks the formal start of the detailed design phase.

Handoff to the CAD team

The RRD is officially delivered to the modeling team, together with:

- An explanation of the robot's functions
- Critical parameters
- Key constraints
- Design priorities

Transition rule

From this point on:

- CAD design strictly follows the RRD
- No informal changes are allowed
- Any deviation goes back to technical evaluation

Change control

If the RRD needs to change during development:

- 1.The change proposal is documented
- 2.The impact on the system is analyzed
- 3.A technical evaluation is run (Go/No-Go style)
- 4.The change is formally approved or rejected

Phase success criteria

The handoff is successful when:

- The CAD team knows exactly what to design
- There are no open questions about functions or constraints
- The process can continue without reinterpretation





DOCUMENTATION

To ensure consistency, traceability and clarity in the information produced during the process, standardized templates are mandatory for documenting each stage.

Consistent formats help to:

- Make communication easier between team members
- Reduce ambiguity
- Speed up content review and validation
- Keep a clear history of decisions

Templates are designed to guide the user to record relevant information in a structured, complete way. Non-standard formats or incomplete documentation compromise the quality of the process and make decisions harder.

Required templates

At a minimum, the following templates are used:

- Rules summary
- A concise document with the key constraints, relevant penalties and strategic opportunities found while analyzing the game manual.
- Prioritization matrix
- A tool to evaluate and rank game tasks by strategic value, considering points, frequency and difficulty.
- Prototype report
- A format to record test results, performance metrics, technical observations and visual evidence of the mechanisms evaluated.
- Robot Requirements Document (RRD)
- A structured template that defines the robot's functionality, parameters, constraints and technology decisions.

Standardization and use

All templates must:

- Be available in a central repository the whole team can access
- Use a uniform structure
- Be mandatory for their corresponding phase
- Stay up to date as the process moves forward





DOCUMENT CONTROL

Documentation produced during the process is managed under basic document control principles to ensure its integrity, currency and traceability.

Required elements

Every document includes, at a minimum:

- Creation date and last modified date
- Author(s) and editor(s)
- Document version
- Change log

This control shows which information is current and how it has evolved over the process.

Version management

We recommend a clear versioning system, for example:

- v1.0: initial version
- v1.1: minor corrections
- v2.0: significant changes

Every relevant change is recorded in the change history, stating:

- What changed
- Why it changed
- Who made the change

Storage and access

Documentation is stored in a central system (e.g. Notion, Google Drive, internal repository) with:

- Role-based access control
- Organization by folders or categories
- Clear, consistent naming

This makes information easy for the whole team to find and use.





PHASE CLOSEOUT

Closing Phase 1 is not based on perception, but on meeting defined criteria that confirm the team is ready for the next stage of the process.

Completion criteria

The phase is complete only when all of the following are met:

- Full understanding of the game
- The whole team shares a clear, aligned interpretation of the game manual with no contradictions.
- Defined, prioritized strategy
- There is a clear ranked task list and a well-established strategic core.
- Validated mechanisms
- The main concepts have been prototyped and evaluated with objective data.
- RRD defined and approved
- The Robot Requirements Document is complete, validated and signed off by the technical leads.
- Complete, organized documentation
- All information from the phase has been recorded, structured and stored correctly.

Final validation

The closeout is validated by:

- Engineering lead
- Design lead
- Strategy captain
- Technical mentor (if applicable)

Once approved, the team is cleared to move on to CAD modeling.

Why closeout matters

A proper closeout ensures that:

- There are no pending or ambiguous decisions
- The team is completely clear on the next step
- Rework in later stages is minimized

