THE MIGHTY CHALLENGE OF MODELLING GEOPOLITICAL BEHAVIOUR IN TOTAL WAR:

HOW AI CAN DELIVER ENTERTAINMENT.

Guy Davidson

@hatcat01

Duygu Cakmak

@aphrael

Overview

- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy
- Tasks and resources
- Profiling and timing

Campaign mechanism

- Factions
- Territory
- Resources

@hatcat01

Campaign mechanism

- Factions
- Territory
- Resources
- Opportunity cost

@hatcat01

Turn sequence

- Inspect resources
- Cities – taxation, construction
- Armies

Turn sequence

- Inspect resources
- Cities – taxation, construction
- Armies
- Agents

Turn sequence

- Now it’s the AI’s turn(s)…
- Attack, besiege, ally, trade, espionage…
- Until…

@hatcat01

Turn sequence

- Now it's the AI's turn(s)…
- Attack, besiege, ally, trade, espionage…
- Until…
- Win conditions

@hatcat01

Win condition

- Control N territories
- Control particular territories
- Eliminate a faction

Win condition

- Control N territories
- Control particular territories
- Eliminate a faction
- Alliances with all factions

@hatcat01

We provide…

- Model of the world
- AI opponent for each faction
- Position and scalar data

We provide…

- Model of the world
- AI opponent for each faction
- Position and scalar data
- Model resolved over time

@hatcat01

Equipping the AI

- Perfect information
- Privileged information

@hatcat01

Equipping the AI

- Perfect information
- Privileged information
- Fun opponent
- Credible opponent

@hatcat01

Equipping the AI

- View API
- Control API

@hatcat01

Equipping the AI

- View API
- Control API
- Motivation

@hatcat01

Equipping the AI

- Personality
- Informed by circumstance
- Traders

Equipping the AI

- Personality
- Informed by circumstance
- Traders
- Fighters

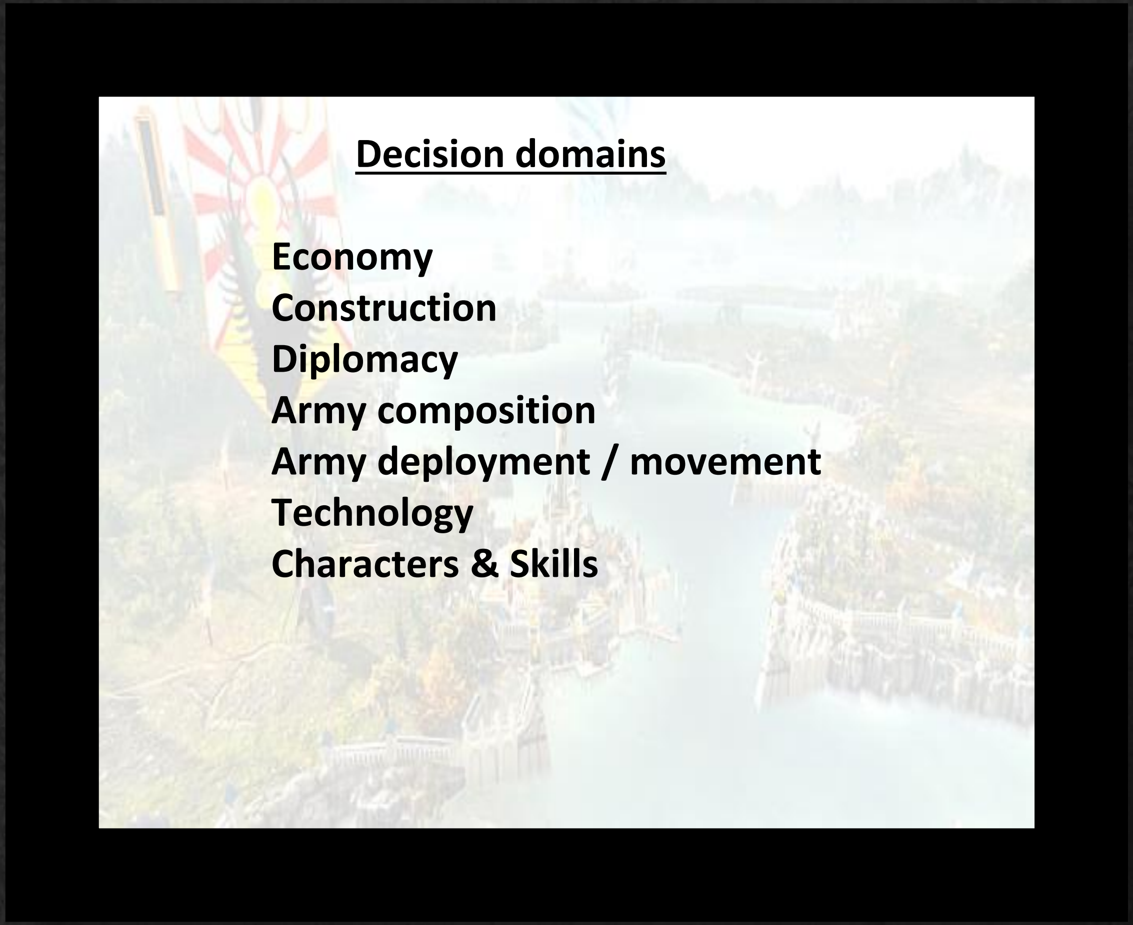
Overview

- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy
- Tasks and resources
- Profiling and timing

@hatcat01

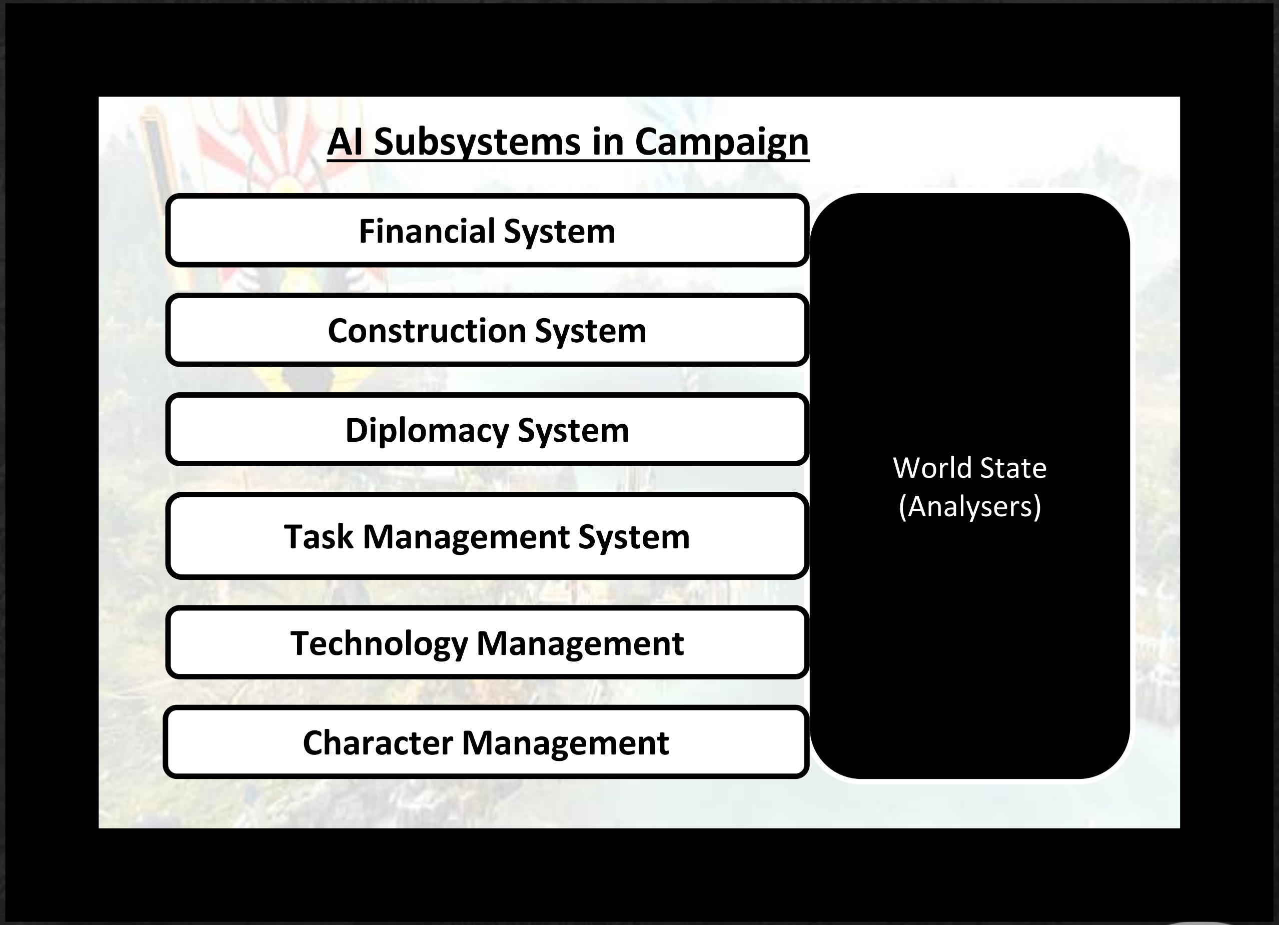
Team effort

- Dr Tim Gosling
- Piotr Andruszkiewicz

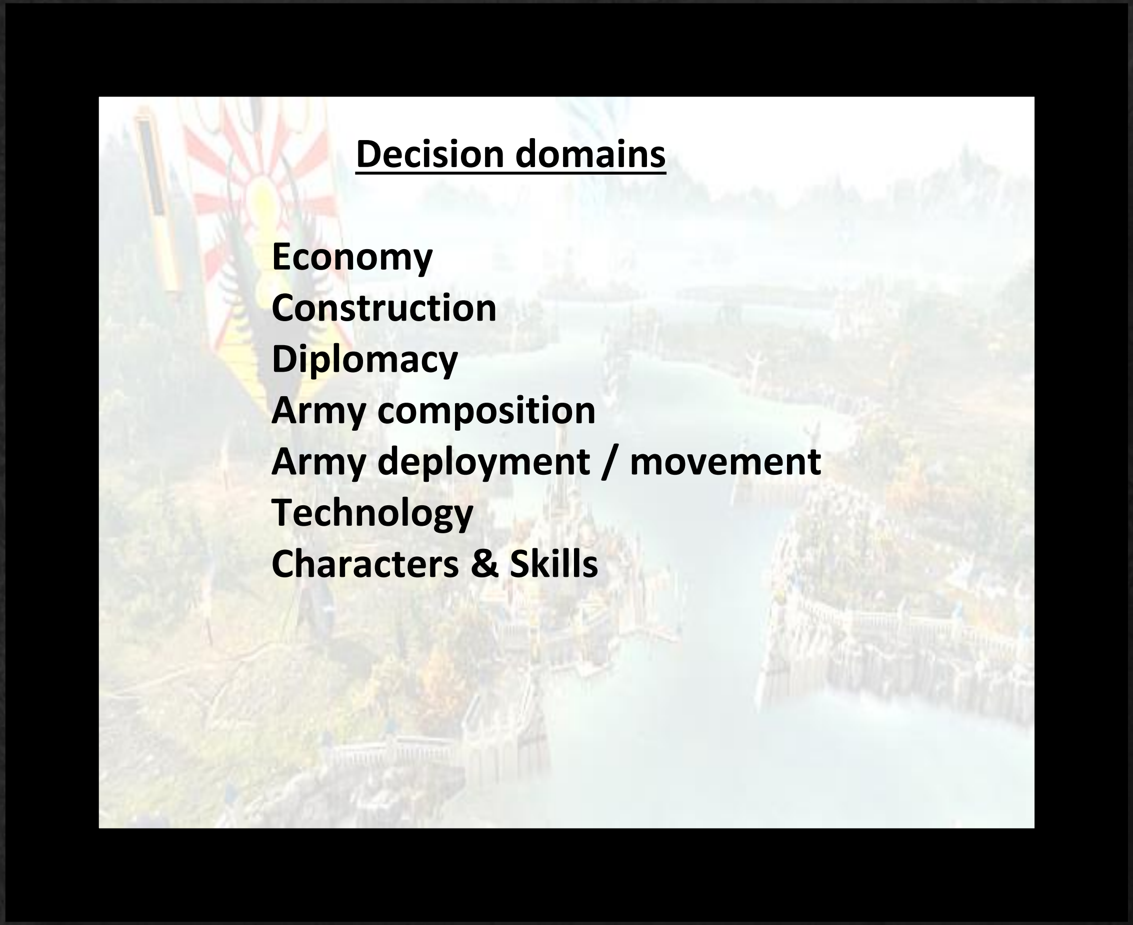


Decision domains

Economy
Construction
Diplomacy
Army composition
Army deployment / movement
Technology
Characters & Skills

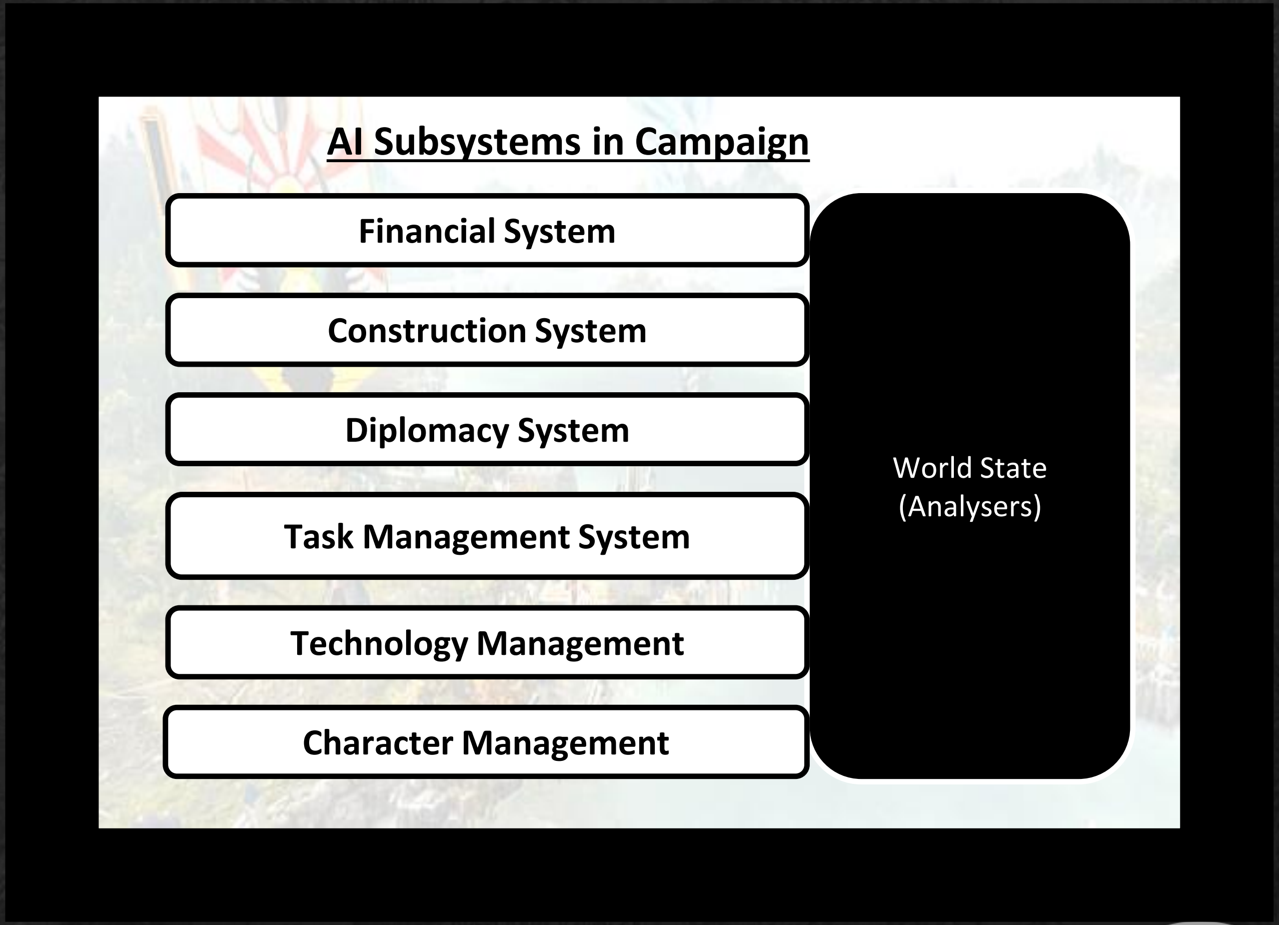


AI Subsystems in Campaign



Decision domains

Economy
Construction
Diplomacy
Army composition
Army deployment / movement
Technology
Characters & Skills

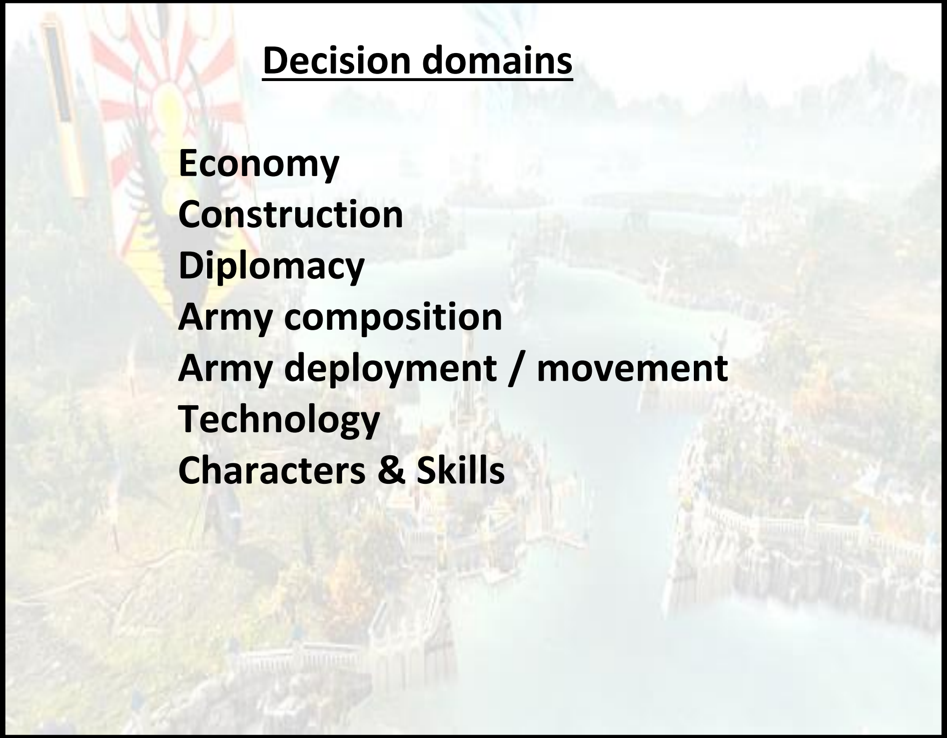


AI Subsystems in Campaign

Financial System
Construction System
Diplomacy System
Task Management System
Technology Management
Character Management

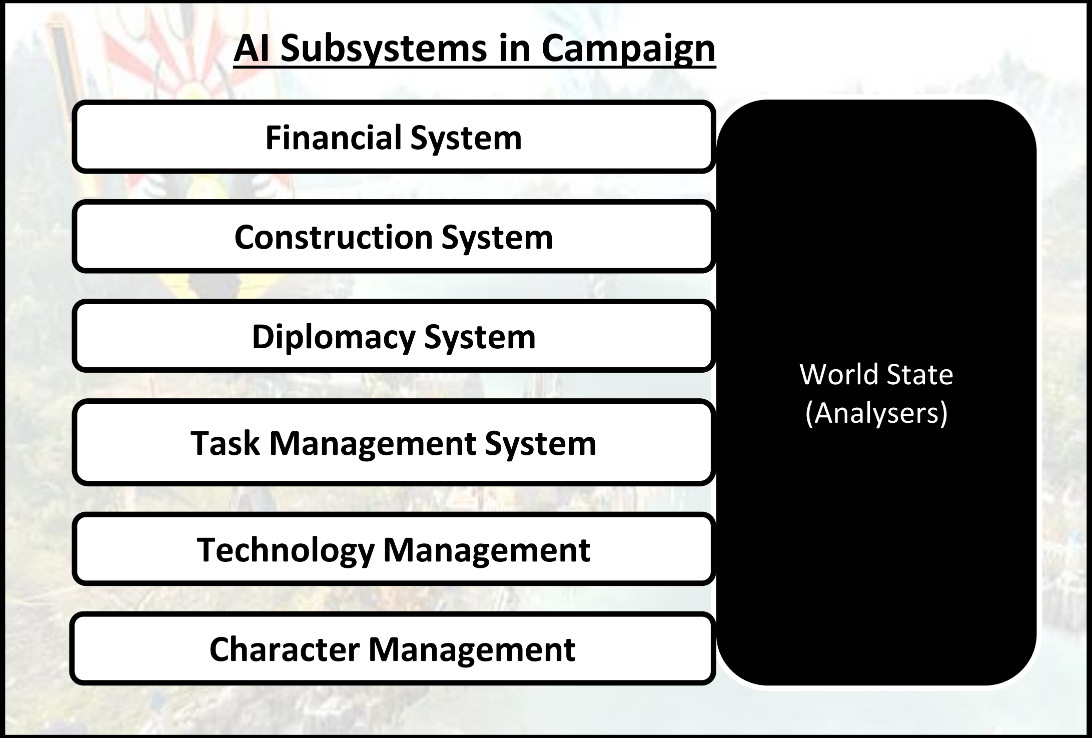
World State
(Analysers)

@aphrael



Decision domains

Economy
Construction
Diplomacy
Army composition
Army deployment / movement
Technology
Characters & Skills

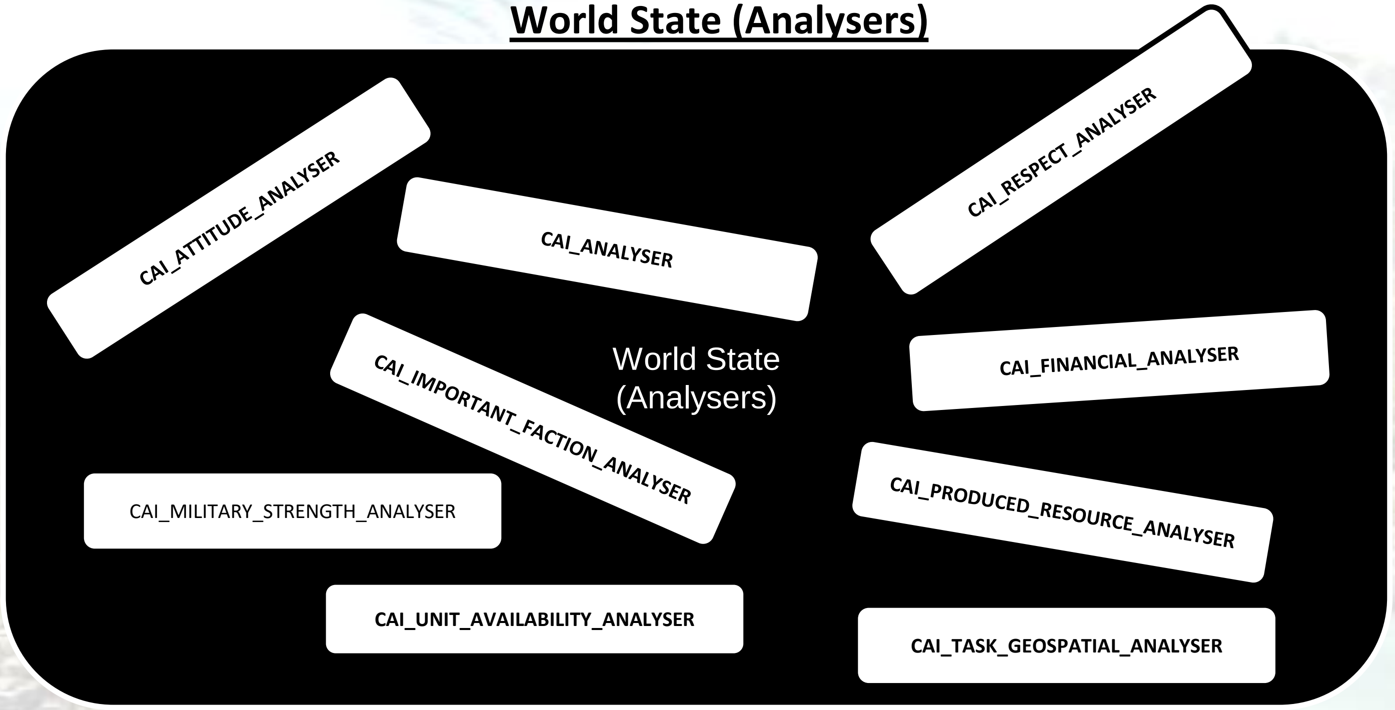


AI Subsystems in Campaign

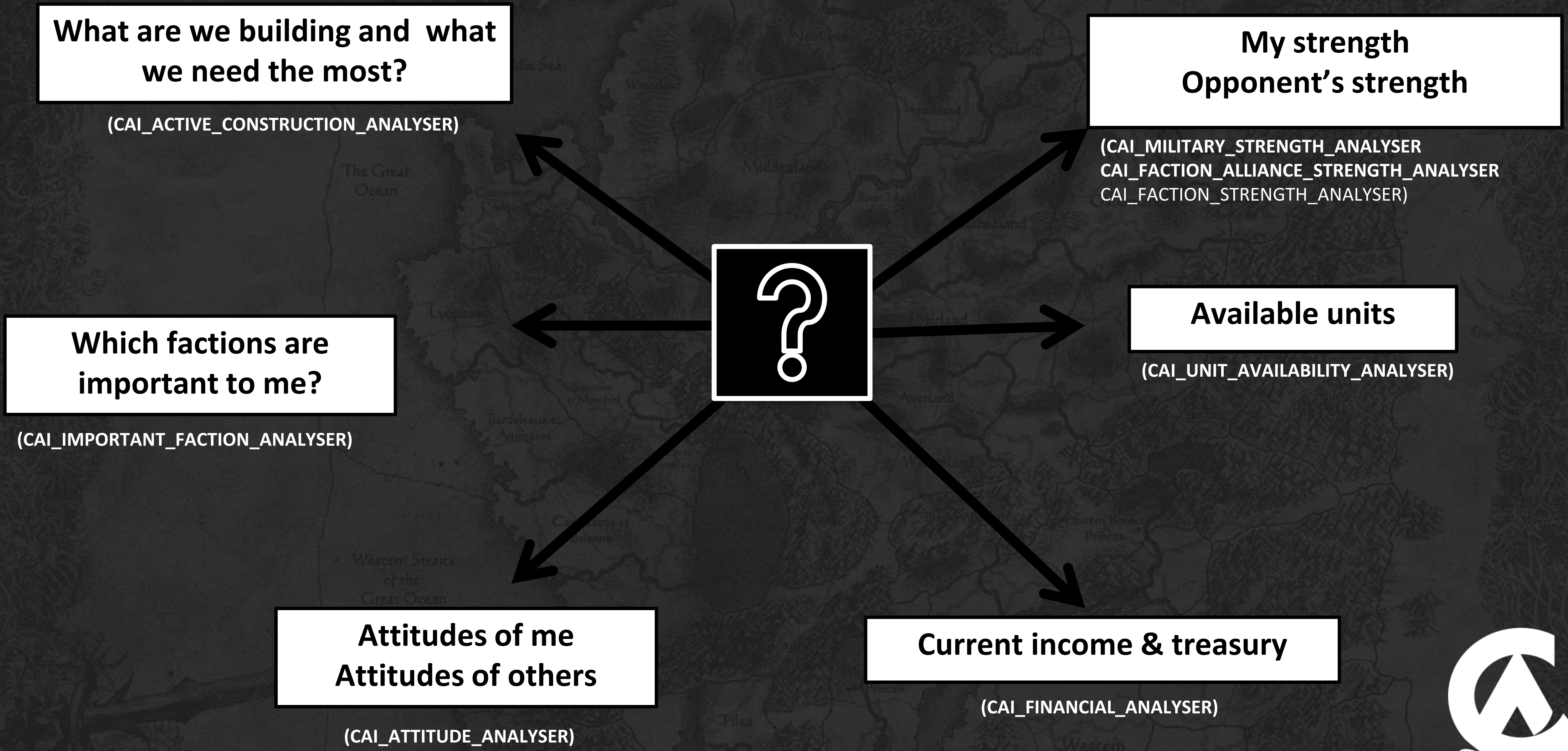
AI Subsystems in Campaign

- Financial System
- Construction System
- Diplomacy System
- Task Management System
- Technology Management
- Character Management

World State (Analysers)


CAI_ATTITUDE_ANALYSER
CAI_ANALYSER
CAI_RESPECT_ANALYSER
World State (Analysers)
CAI_FINANCIAL_ANALYSER
CAI_IMPORTANT_FACTION_ANALYSER
CAI_MILITARY_STRENGTH_ANALYSER
CAI_PRODUCED_RESOURCE_ANALYSER
CAI_UNIT_AVAILABILITY_ANALYSER
CAI_TASK_GEOSPATIAL_ANALYSER

@aphrael


What are we building and what we need the most?
(CAI_ACTIVE_CONSTRUCTION_ANALYSER)
My strength
Opponent's strength
(CAI_MILITARY_STRENGTH_ANALYSER
CAI_FACTION_ALLIANCE_STRENGTH_ANALYSER
CAI_FACTION_STRENGTH_ANALYSER)
?
Which factions are important to me?
(CAI_IMPORTANT_FACTION_ANALYSER)
Available units
(CAI_UNIT_AVAILABILITY_ANALYSER)
Attitudes of me
Attitudes of others
(CAI_ATTITUDE_ANALYSER)
Current income & treasury
(CAI_FINANCIAL_ANALYSER)

What are we building and what we need the most?

(CAI_ACTIVE_CONSTRUCTION_ANALYSER)

My strength
Opponent's strength

(CAI_MILITARY_STRENGTH_ANALYSER
CAI_FACTION_ALLIANCE_STRENGTH_ANALYSER
CAI_FACTION_STRENGTH_ANALYSER)

Which factions are important to me?

(CAI_IMPORTANT_FACTION_ANALYSER)

Available units

(CAI_UNIT_AVAILABILITY_ANALYSER)

Attitudes of me
Attitudes of others

(CAI_ATTITUDE_ANALYSER)

Current income & treasury

(CAI_FINANCIAL_ANALYSER)

What are we building and what we need the most?

(CAI_ACTIVE_CONSTRUCTION_ANALYSER)

My strength
Opponent's strength

(CAI_MILITARY_STRENGTH_ANALYSER
CAI_FACTION_ALLIANCE_STRENGTH_ANALYSER
CAI_FACTION_STRENGTH_ANALYSER)

Which factions are important to me?

(CAI_IMPORTANT_FACTION_ANALYSER)

Available units

(CAI_UNIT_AVAILABILITY_ANALYSER)

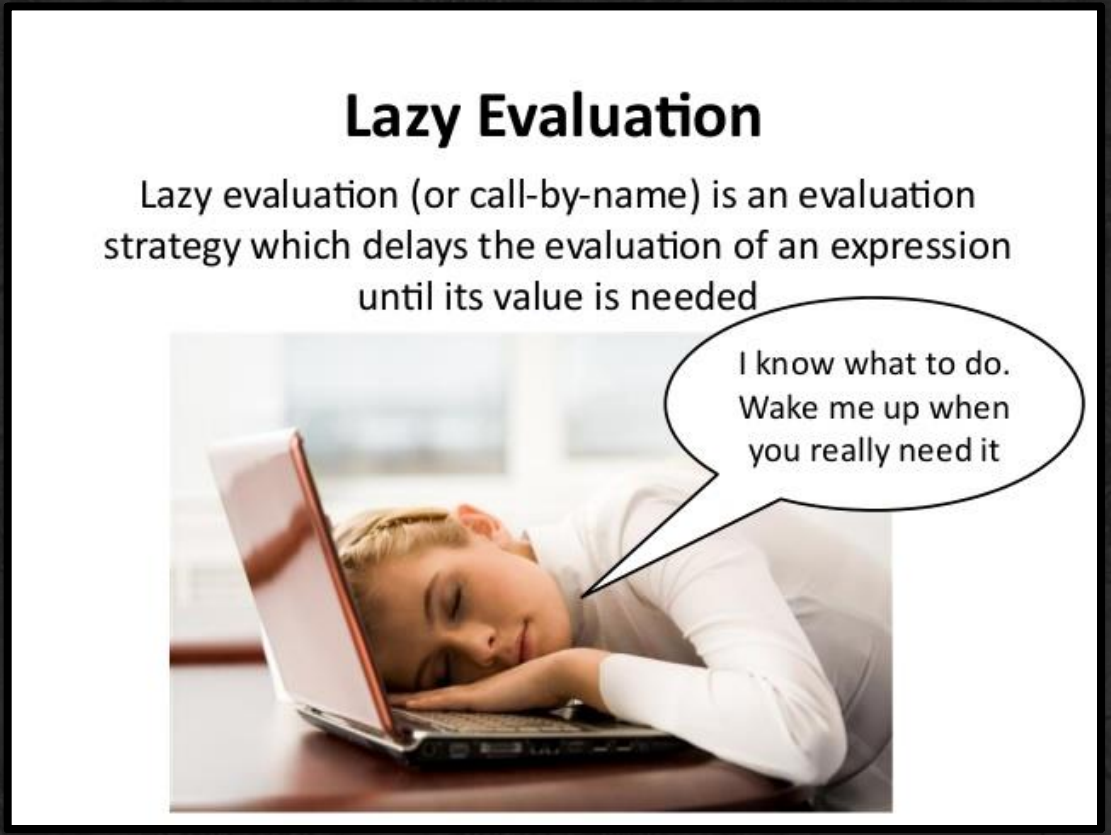
Attitudes of me
Attitudes of others

(CAI_ATTITUDE_ANALYSER)

Current income & treasury

(CAI_FINANCIAL_ANALYSER)

Analysers

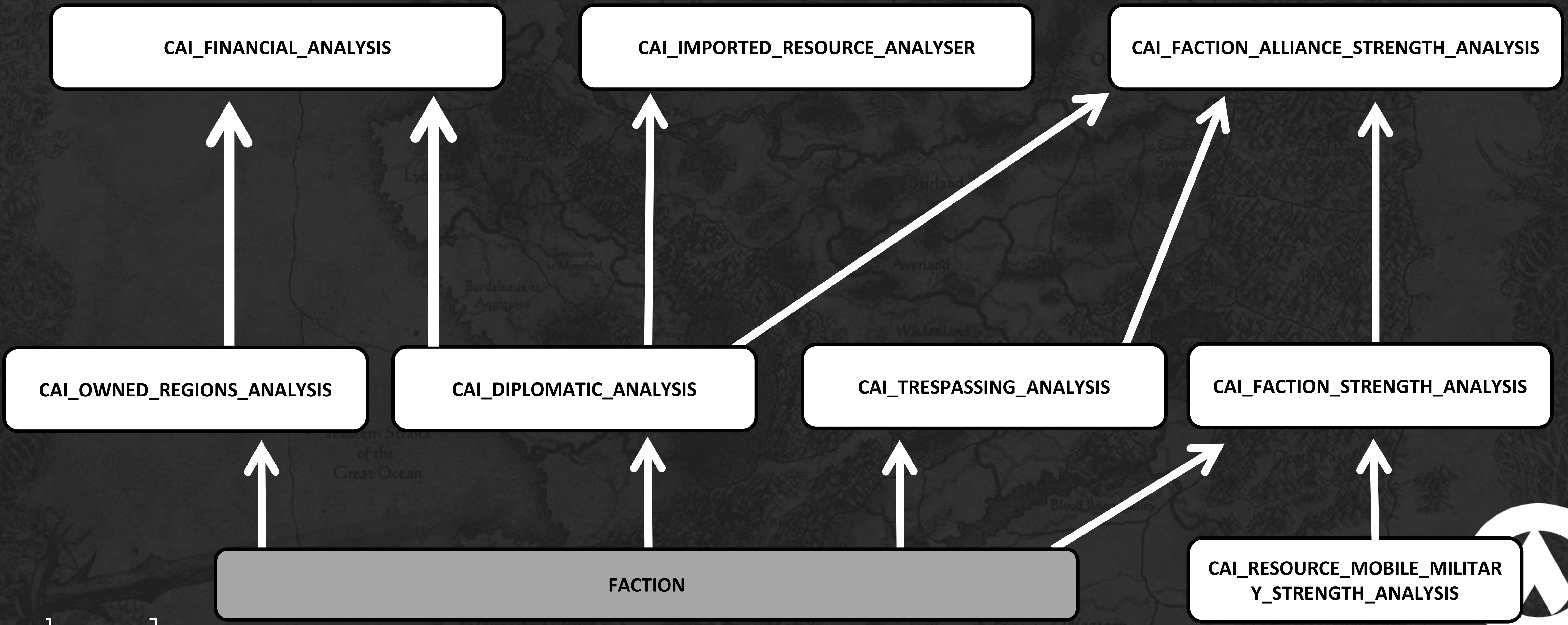

Lazy Evaluation

Lazy evaluation (or call-by-name) is an evaluation strategy which delays the evaluation of an expression until its value is needed

I know what to do. Wake me up when you really need it

@aphrael

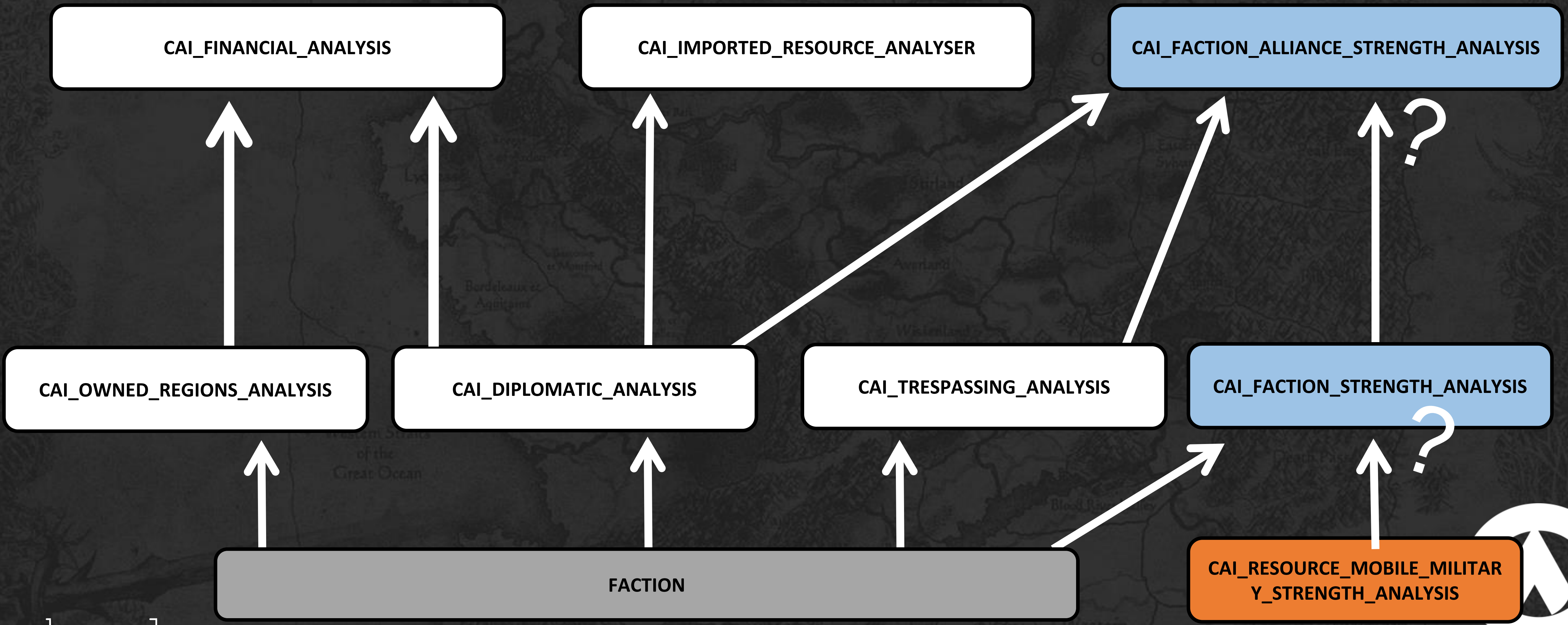
Analysers



CAI_FINANCIAL_ANALYSIS
CAI_IMPORTED_RESOURCE_ANALYSER
CAI_FACTION_ALLIANCE_STRENGTH_ANALYSIS
CAI_OWNED_REGIONS_ANALYSIS
CAI_DIPLOMATIC_ANALYSIS
CAI_TRESPASSING_ANALYSIS
CAI_FACTION_STRENGTH_ANALYSIS
FACTION
CAI_RESOURCE_MOBILE_MILITARY_STRENGTH_ANALYSIS

@aphrael

Analysers


CAI_FINANCIAL_ANALYSIS
CAI_IMPORTED_RESOURCE_ANALYSER
CAI_FACTION_ALLIANCE_STRENGTH_ANALYSIS
?
CAI_OWNED_REGIONS_ANALYSIS
CAI_DIPLOMATIC_ANALYSIS
CAI_TRESPASSING_ANALYSIS
CAI_FACTION_STRENGTH_ANALYSIS
?
FACTION
CAI_RESOURCE_MOBILE_MILITARY_STRENGTH_ANALYSIS

@aphrael

Analysers

$$\begin{tikzcd}
\text{CAI\_ANALYSER} \\
\text{CAI\_ANALYSER\_BASE} \arrow[u]
\end{tikzcd}$$

CAI_ACTIVE_CONSTRUCTION_ANALYSER
CAI_ACTIVE_RECRUITMENT_ANALYSER
CAI_ATTITUDE_ANALYSER
CAI_ATTRITION_ANALYSER
CAI_BASIC_REGION_GROUP_ANALYSER
CAI_BUILDING_AVAILABILITY_ANALYSER
CAI_CHARACTER_ROLE_ANALYSER
CAI_ANALYSER
CAI_DIRECT_ATTITUDE_ANALYSER
CAI_FACTION_ALLIANCE_STRENGTH_ANALYSER
CAI_FACTION_RESEARCH_TECHNOLOGY_ANALYSER
CAI_FACTION_STRENGTH_ANALYSER
CAI_FACTION_TAXATION_ANALYSER
CAI_FACTIONWIDE_UNIT_AVAILABILITY_ANALYSER
CAI_FERTILITY_ANALYSER
CAI_FINANCIAL_ANALYSER

CAI_FOOD_ANALYSER
CAI_IMPORTANT_FACTION_ANALYSER
CAI_IMPORTED_RESOURCE_ANALYSER
CAI_MILITARY_ACCESS_ANALYSER
CAI_MILITARY_STRENGTH_ANALYSER
CAI_NEIGHBOURING_FACTIONS_ANALYSER
CAI_OWNED_REGIONS_ANALYSER
CAI_PLAYER_PROXIMITY_ANALYSER
CAI_PRODUCED_RESOURCE_ANALYSER
CAI_RESPECT_ANALYSER
CAI_STRATEGIC_CONTEXT_ANALYSER
CAI_TASK_GEOSPATIAL_ANALYSER
CAI_TASK_RECRUITMENT_PREFERENCE_ANALYSER
CAI_TRESPASSING_ANALYSER
CAI_UNIT_AVAILABILITY_ANALYSER
CAI_VICTORY_REGION_ANALYSER

CAI_ANALYSER_BASE

```
template<class ANALYSED_CLASS, class ANALYSIS_CLASS, BDI_CLASSES_IDENTIFICATION analyser_id>
class CAI_ANALYSER_BASE : public CAI_ANALYSER
{

public:
        CAI_ANALYSER_BASE                                         (CAI_BDI_POOL &bdi_pool);
        virtual bool            post_load_fix_up                  (CAI_BDI_POOL &bdi_pool);
        virtual bool            validate                          (CAI_BDI_POOL &bdi_pool);
        virtual void            save                              (EmpireFileOutSection &saveto) const;

        ANALYSIS_CLASS &        determine_analysis_for            (ANALYSED_CLASS & target);
        const ANALYSIS_CLASS *  determine_analysis_for_const(const ANALYSED_CLASS & target) const;

};
```

@aphrael

CAI_ANALYSER_BASE::determine_analysis_for

```
template<class ANALYSED_CLASS, class ANALYSIS_CLASS, BDI_CLASSES_IDENTIFICATION analyser_id>
ANALYSIS_CLASS &CAI_ANALYSER_BASE<ANALYSED_CLASS, ANALYSIS_CLASS, analyser_id>
    ::determine_analysis_for(ANALYSED_CLASS & target)
{
    CAI_ANALYSIS *general_analysis = get_analysis(target.get_bdi_index());
         ANALYSIS_CLASS *specific_analysis = nullptr;
         if( general_analysis )
         {
                  specific_analysis = static_cast<ANALYSIS_CLASS *>(general_analysis);
         }
         else
         {
                  specific_analysis = new ANALYSIS_CLASS(target);
                  add_analysis(*specific_analysis, target.get_bdi_index());
         }
         if( specific_analysis->is_invalidated() )
         {
                  specific_analysis->do_validation(get_bdi_pool());
         }
         return *specific_analysis;
}
```

@aphrael

CAI_ANALYSER_BASE::determine_analysis_for

```
template<class ANALYSED_CLASS, class ANALYSIS_CLASS, BDI_CLASSES_IDENTIFICATION analyser_id>
ANALYSIS_CLASS &CAI_ANALYSER_BASE<ANALYSED_CLASS, ANALYSIS_CLASS, analyser_id>
    ::determine_analysis_for(ANALYSED_CLASS & target)
{
    CAI_ANALYSIS *general_analysis = get_analysis(target.get_bdi_index());
         ANALYSIS_CLASS *specific_analysis = nullptr;
         if( general_analysis )
         {
                  specific_analysis = static_cast<ANALYSIS_CLASS *>(general_analysis);
         }
         else
         {
                  specific_analysis = new ANALYSIS_CLASS(target);
                  add_analysis(*specific_analysis, target.get_bdi_index());
         }
         if( specific_analysis->is_invalidated() )
         {
                  specific_analysis->do_validation(get_bdi_pool());
         }
         return *specific_analysis;
}
```

@aphrael

CAI_ANALYSER_BASE::determine_analysis_for

```cpp
template<class ANALYSED_CLASS, class ANALYSIS_CLASS, BDI_CLASSES_IDENTIFICATION analyser_id>
ANALYSIS_CLASS &CAI_ANALYSER_BASE<ANALYSED_CLASS, ANALYSIS_CLASS, analyser_id>
    ::determine_analysis_for(ANALYSED_CLASS & target)
{
    CAI_ANALYSIS *general_analysis = get_analysis(target.get_bdi_index());
         ANALYSIS_CLASS *specific_analysis = nullptr;
         if( general_analysis )
         {
                  specific_analysis = static_cast<ANALYSIS_CLASS *>(general_analysis);
         }
         else
         {
                  specific_analysis = new ANALYSIS_CLASS(target);
                  add_analysis(*specific_analysis, target.get_bdi_index());
         }
         if( specific_analysis->is_invalidated() )
         {
                  specific_analysis->do_validation(get_bdi_pool());
         }
         return *specific_analysis;
}
```

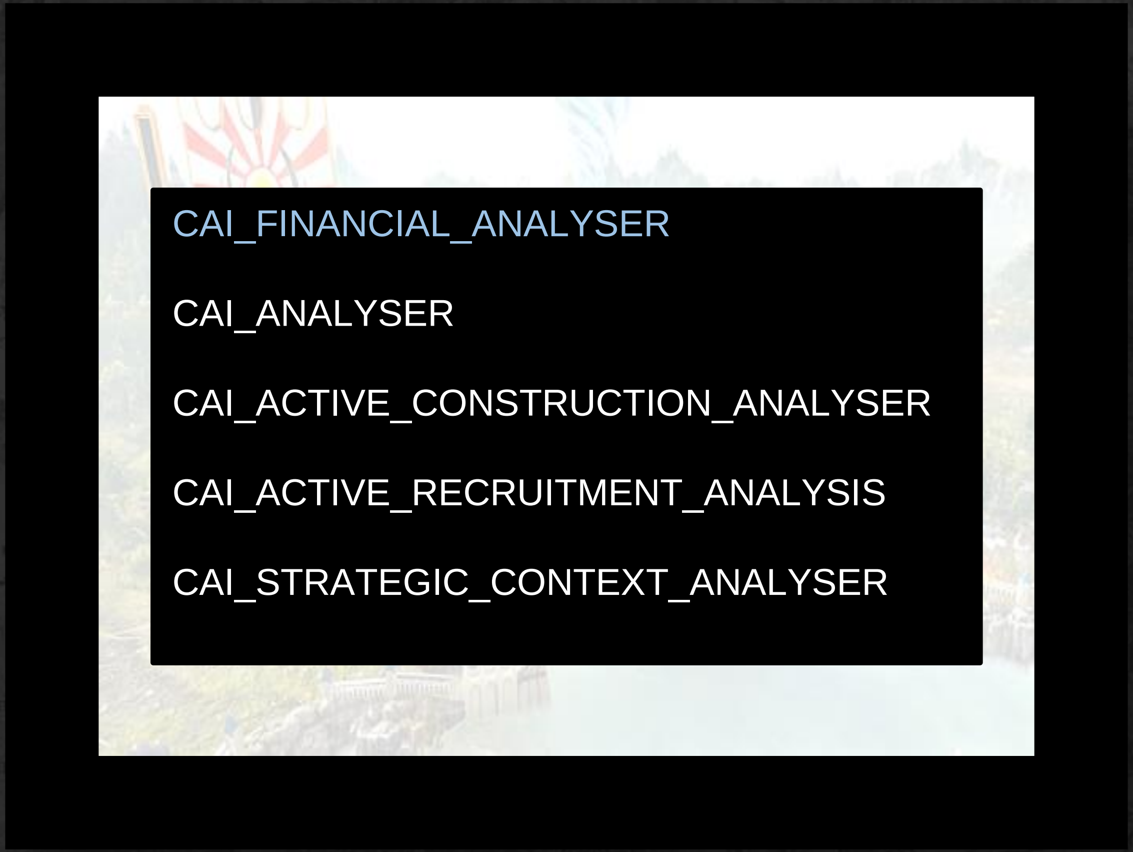
@aphrael

CAI_ANALYSER_BASE::determine_analysis_for

```
template<class ANALYSED_CLASS, class ANALYSIS_CLASS, BDI_CLASSES_IDENTIFICATION analyser_id>
ANALYSIS_CLASS &CAI_ANALYSER_BASE<ANALYSED_CLASS, ANALYSIS_CLASS, analyser_id>
    ::determine_analysis_for(ANALYSED_CLASS & target)
{
    CAI_ANALYSIS *general_analysis = get_analysis(target.get_bdi_index());
         ANALYSIS_CLASS *specific_analysis = nullptr;
         if( general_analysis )
         {
                  specific_analysis = static_cast<ANALYSIS_CLASS *>(general_analysis);
         }
         else
         {
                  specific_analysis = new ANALYSIS_CLASS(target);
                  add_analysis(*specific_analysis, target.get_bdi_index());
         }
         if( specific_analysis->is_invalidated() )
         {
                  specific_analysis->do_validation(get_bdi_pool());
         }
         return *specific_analysis;
}
```

@aphrael

Financial system



CAI_FINANCIAL_ANALYSER

CAI_ANALYSER

CAI_ACTIVE_CONSTRUCTION_ANALYSER

CAI_ACTIVE_RECRUITMENT_ANALYSIS

CAI_STRATEGIC_CONTEXT_ANALYSER

@aphrael

CAI_FINANCIAL_ANALYSIS

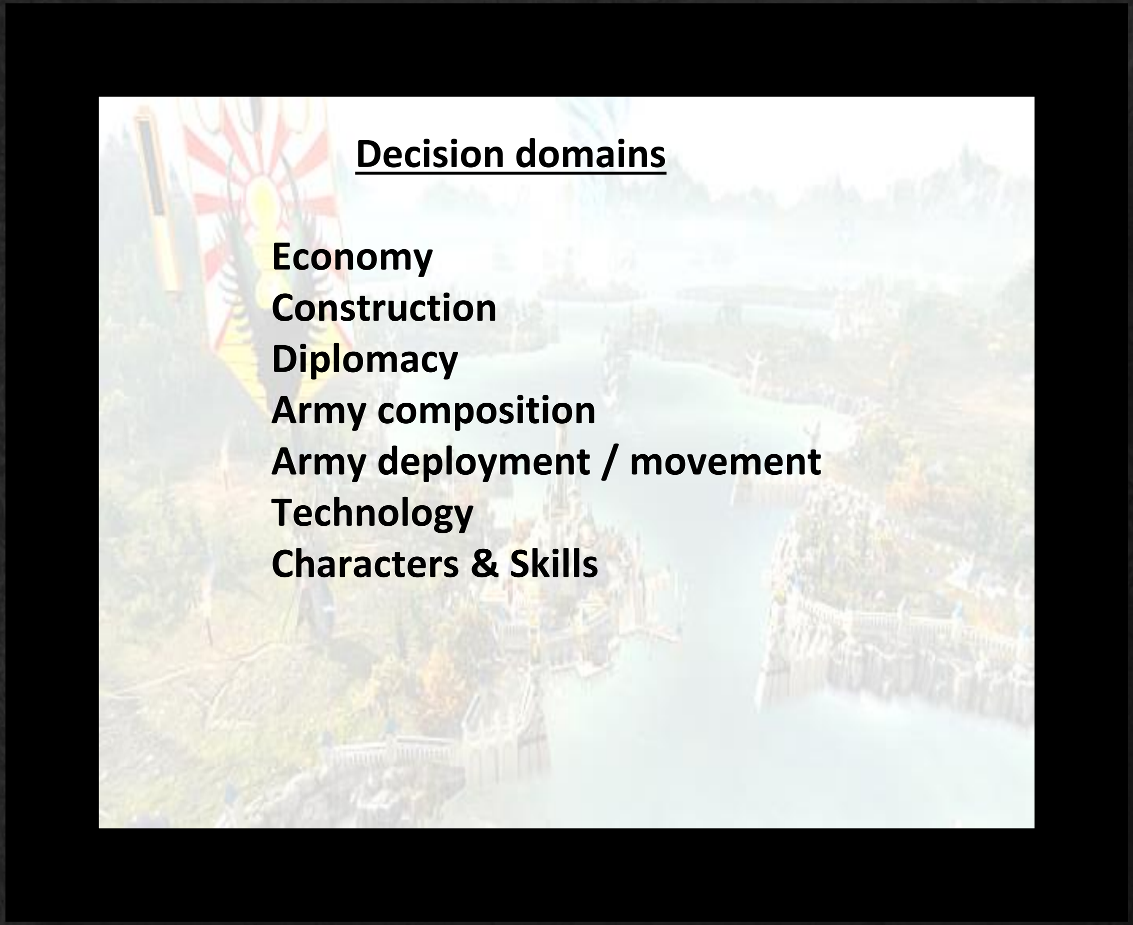
```
class CAI_FINANCIAL_ANALYSIS : public CAI_ANALYSIS
{
public:
CAI_FINANCIAL_ANALYSIS                         (CAI_FACTION & faction);
virtual void on_pool_add                       (CAI_BDI_POOL & bdi_pool);

// Information
float32   currently_acceptable_balance         () const;
float32   absolute_acceptable_turn_on_outgoings() const;
void      calculate_spending_value             (CAI_BDI_POOL & bdi_pool,
                                                float32 &     relative_acceptable_balance,
                                                float32  &     absolute_turn_on_turn_outgoings,
                                                CAI_FACTION *  optional_additional_faction_to_consider_at_war);

private:
CONST_SAFE_PTR<CAI_FACTION> m_faction;
Float32                     m_currently_acceptable_balance;
Float32                     m_absolute_acceptable_turn_on_turn_outgoings;

virtual bool validate                          (CAI_BDI_POOL & bdi_pool); // Affect: True if things have
changed as a result
};
```

@aphrael

CAI_FINANCIAL_ANALYSIS::validate

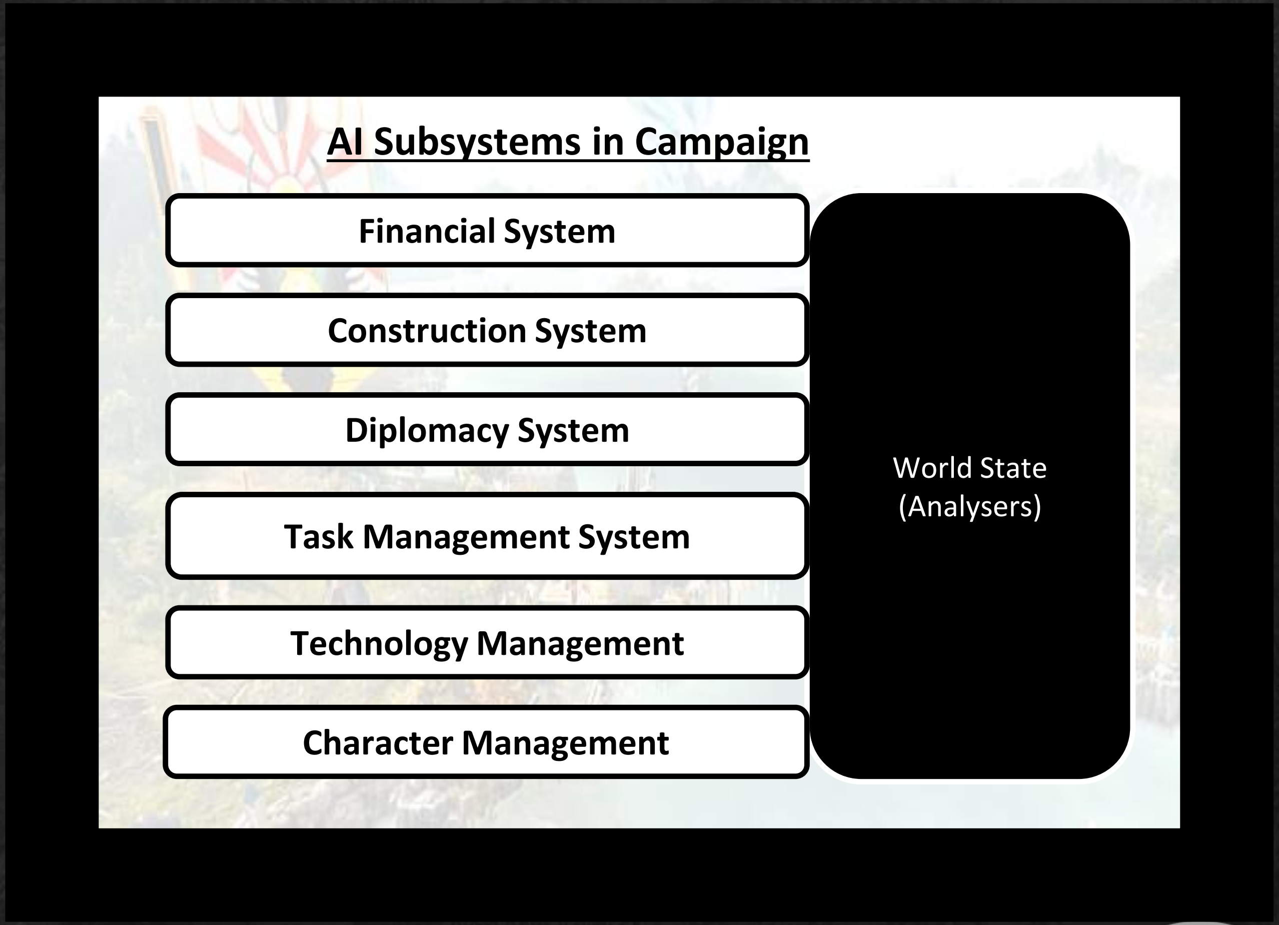
```
bool CAI_FINANCIAL_ANALYSIS::validate(CAI_BDI_POOL &bdi_pool)
{
         // Force update of analysis
         bdi_pool.get_central_bdi_pool().owned_regions_analyser().determine_analysis_for(*m_faction);

         // Generate new information
         float32 temp_selected_acceptable_balance     = 0.0f;
         float32 temp_absolute_turn_on_turn_outgoings = 0.0f;

          calculate_spending_value(bdi_pool, temp_selected_acceptable_balance,
                                              temp_absolute_turn_on_turn_outgoings, null);

         // Update if needed
         if( temp_selected_acceptable_balance != m_currently_acceptable_balance ||
         temp_absolute_turn_on_turn_outgoings != m_absolute_acceptable_turn_on_turn_outgoings )
         {
                  m_currently_acceptable_balance                = temp_selected_acceptable_balance;
                  m_absolute_acceptable_turn_on_turn_outgoings = temp_absolute_turn_on_turn_outgoings;
                  return true;
         }
         return false;
}
```

@aphrael



Decision domains

Economy
Construction
Diplomacy
Army composition
Army deployment / movement
Technology
Characters & Skills

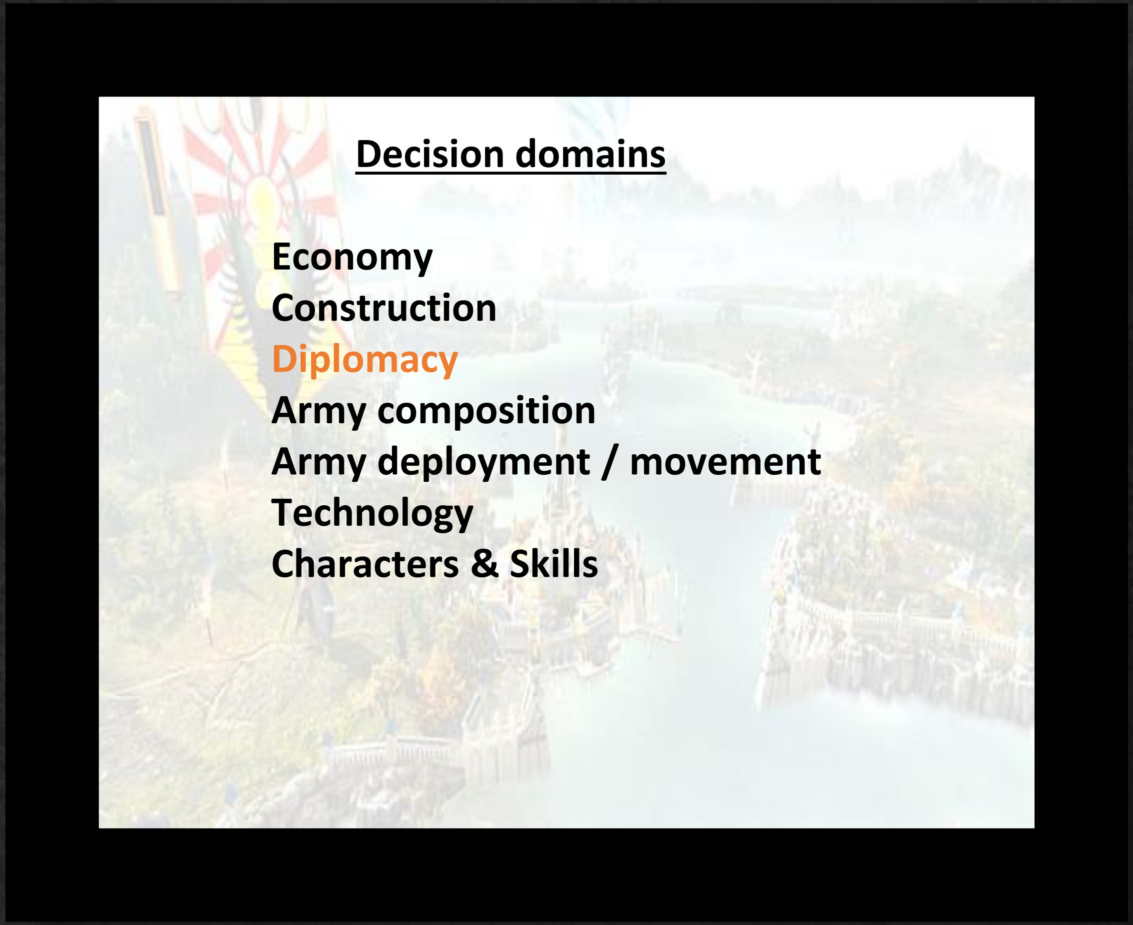


AI Subsystems in Campaign

@aphrael

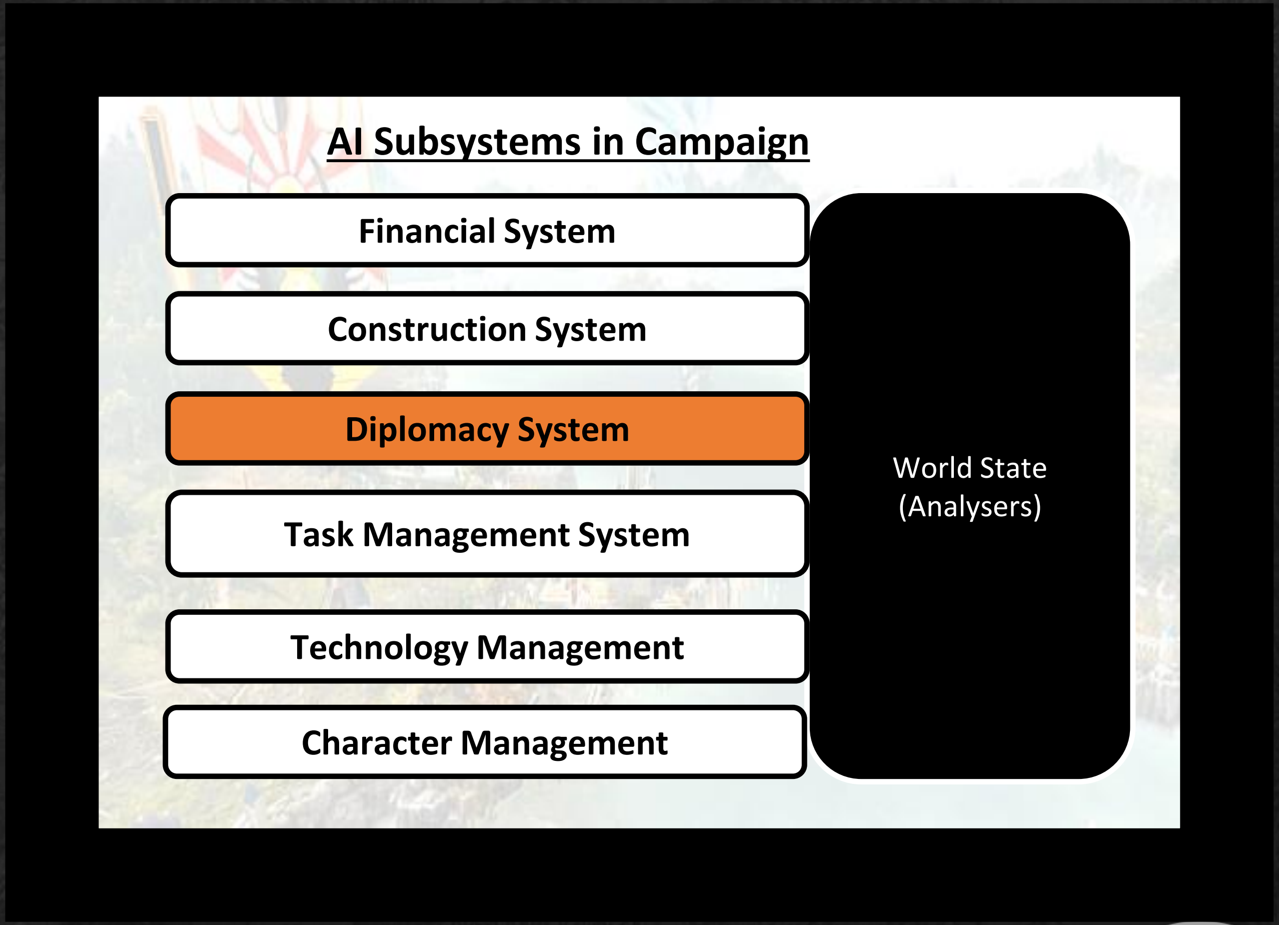
Overview

- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy
- Tasks and resources
- Profiling and timing



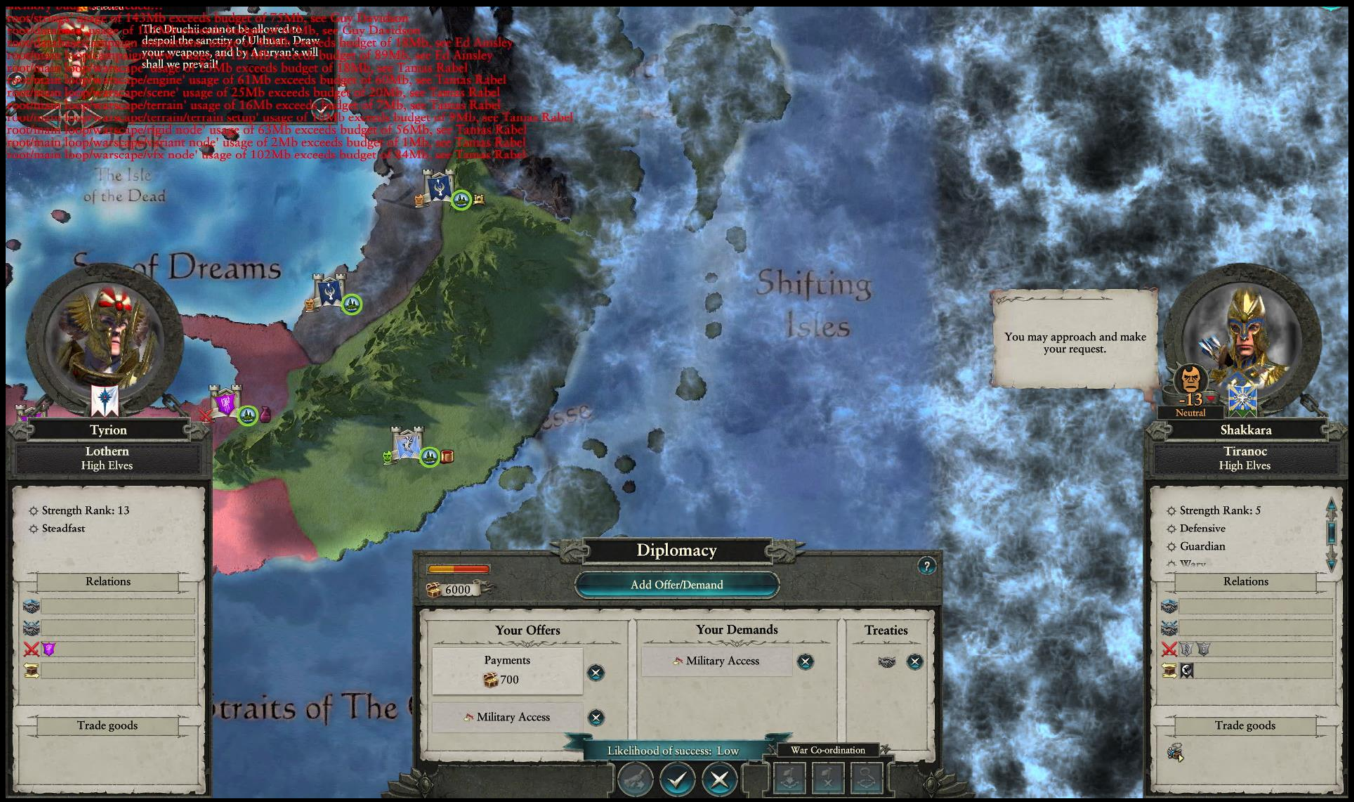
Decision domains

Economy
Construction
Diplomacy
Army composition
Army deployment / movement
Technology
Characters & Skills



AI Subsystems in Campaign

Diplomacy system



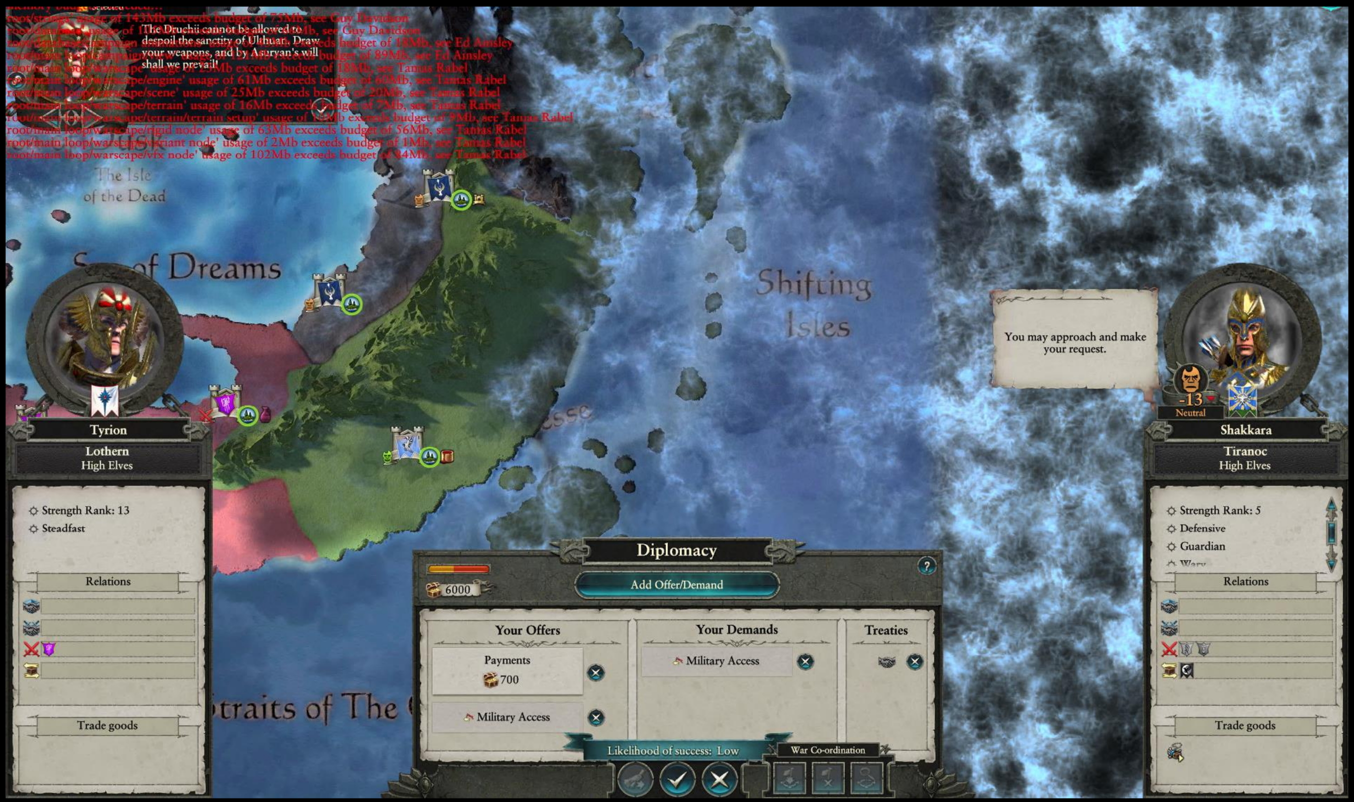
```
memory budget exceeded...
root/strings' usage of 143Mb exceeds budget of 75Mb, see Guy Davidson
root/[illegible] usage of [illegible] exceeds budget of [illegible]Mb, see Guy Davidson
root/database/campaign animations' usage of [illegible]Mb exceeds budget of 18Mb, see Ed Ainsley
root/main loop/campaign view' usage of [illegible]Mb exceeds budget of 89Mb, see Ed Ainsley
root/main loop/warscape' usage of 25Mb exceeds budget of 18Mb, see Tamas Rabel
root/main loop/warscape/engine' usage of 61Mb exceeds budget of 60Mb, see Tamas Rabel
root/main loop/warscape/scene' usage of 25Mb exceeds budget of 20Mb, see Tamas Rabel
root/main loop/warscape/terrain' usage of 16Mb exceeds budget of 7Mb, see Tamas Rabel
root/main loop/warscape/terrain/terrain setup' usage of 10Mb exceeds budget of 9Mb, see Tamas Rabel
root/main loop/warscape/rigid node' usage of 63Mb exceeds budget of 56Mb, see Tamas Rabel
root/main loop/warscape/variant node' usage of 2Mb exceeds budget of 1Mb, see Tamas Rabel
root/main loop/warscape/vfx node' usage of 102Mb exceeds budget of 84Mb, see Tamas Rabel
```

The Druchii cannot be allowed to despoil the sanctity of Ulthuan. Draw your weapons, and by Asuryan's will shall we prevail!

You may approach and make your request.

@aphrael

Diplomacy system

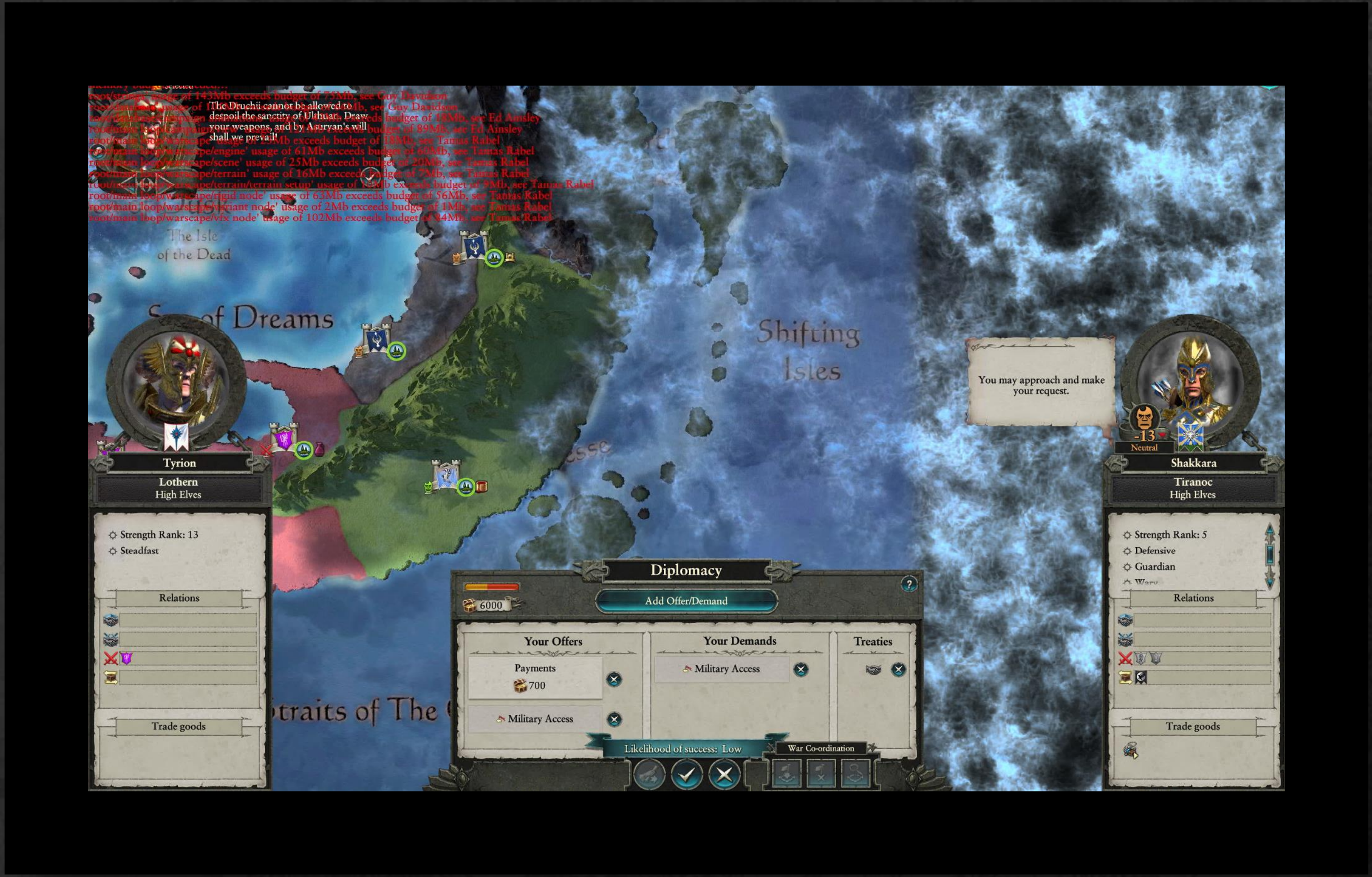


```
memory budget exceeded...
root/strings' usage of 143Mb exceeds budget of 75Mb, see Guy Davidson
root/[illegible] usage of [illegible] exceeds budget of [illegible]Mb, see Guy Davidson
root/database/campaign animations' usage of [illegible] exceeds budget of 18Mb, see Ed Ainsley
root/main loop/campaign view' usage of [illegible]Mb exceeds budget of 89Mb, see Ed Ainsley
root/main loop/warscape' usage of 2[illegible]Mb exceeds budget of 18Mb, see Tamas Rabel
root/main loop/warscape/engine' usage of 61Mb exceeds budget of 60Mb, see Tamas Rabel
root/main loop/warscape/scene' usage of 25Mb exceeds budget of 20Mb, see Tamas Rabel
root/main loop/warscape/terrain' usage of 16Mb exceeds budget of 7Mb, see Tamas Rabel
root/main loop/warscape/terrain/terrain setup' usage of 1[illegible]Mb exceeds budget of 9Mb, see Tamas Rabel
root/main loop/warscape/rigid node' usage of 63Mb exceeds budget of 56Mb, see Tamas Rabel
root/main loop/warscape/variant node' usage of 2Mb exceeds budget of 1Mb, see Tamas Rabel
root/main loop/warscape/vfx node' usage of 102Mb exceeds budget of 84Mb, see Tamas Rabel
```

The Druchii cannot be allowed to despoil the sanctity of Ulthuan. Draw your weapons, and by Asuryan's will shall we prevail!

You may approach and make your request.

Diplomacy system



```
root/strings' usage of 143Mb exceeds budget of 75Mb, see Guy Davidson
root/[illegible] usage of [illegible]Mb, see Guy Davidson
root/database/campaign [illegible] exceeds budget of 18Mb, see Ed Ainsley
root/main loop/campaign [illegible] exceeds budget of 89Mb, see Ed Ainsley
root/main loop/warscape' usage of 25Mb exceeds budget of 18Mb, see Tamas Rabel
root/main loop/warscape/engine' usage of 61Mb exceeds budget of 60Mb, see Tamas Rabel
root/main loop/warscape/scene' usage of 25Mb exceeds budget of 20Mb, see Tamas Rabel
root/main loop/warscape/terrain' usage of 16Mb exceeds budget of 7Mb, see Tamas Rabel
root/main loop/warscape/terrain/terrain setup' usage of 10Mb exceeds budget of 9Mb, see Tamas Rabel
root/main loop/warscape/rigid node' usage of 63Mb exceeds budget of 56Mb, see Tamas Rabel
root/main loop/warscape/variant node' usage of 2Mb exceeds budget of 1Mb, see Tamas Rabel
root/main loop/warscape/vfx node' usage of 102Mb exceeds budget of 84Mb, see Tamas Rabel
```

The Druchii cannot be allowed to despoil the sanctity of Ulthuan. Draw your weapons, and by Asuryan's will shall we prevail!

You may approach and make your request.

Tyrion
Lothern
High Elves

- Strength Rank: 13
- Steadfast

Shakkara
Tiranoc
High Elves
-13 Neutral

- Strength Rank: 5
- Defensive
- Guardian

Diplomacy

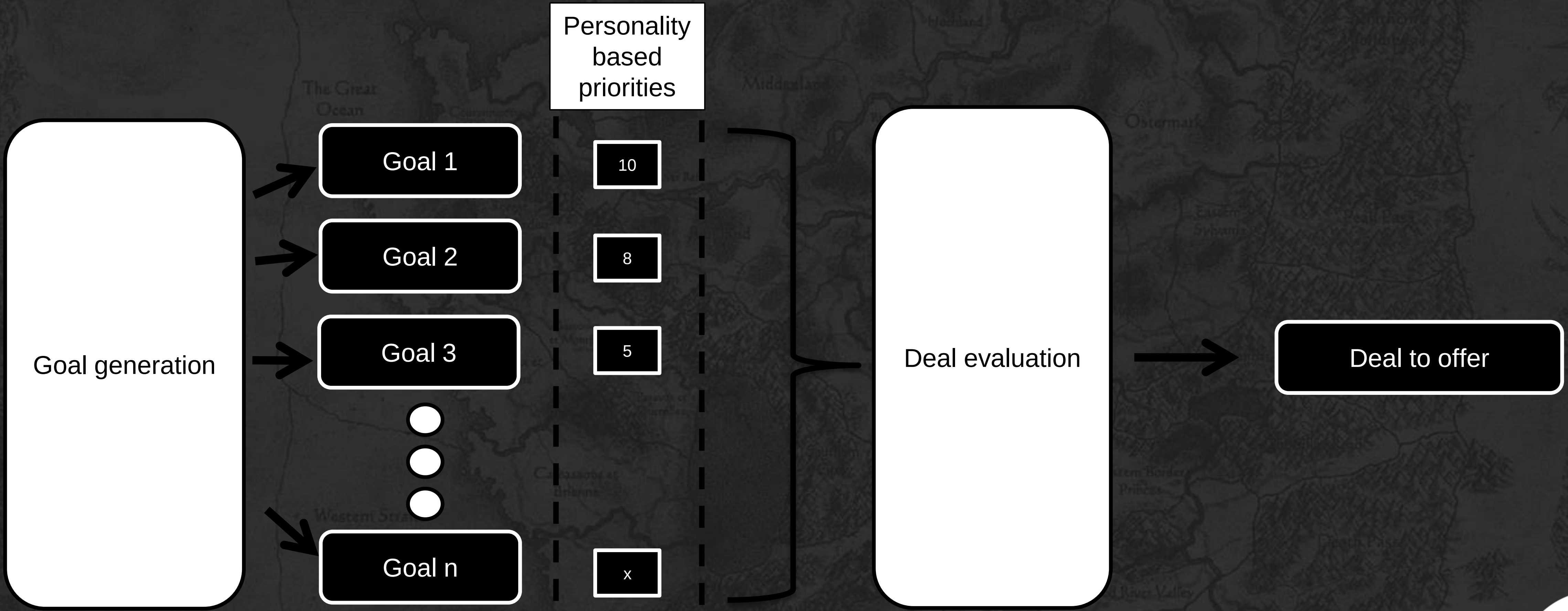
6000

Your Offers	Your Demands	Treaties
Payments 700	Military Access	
Military Access		

Likelihood of success: Low

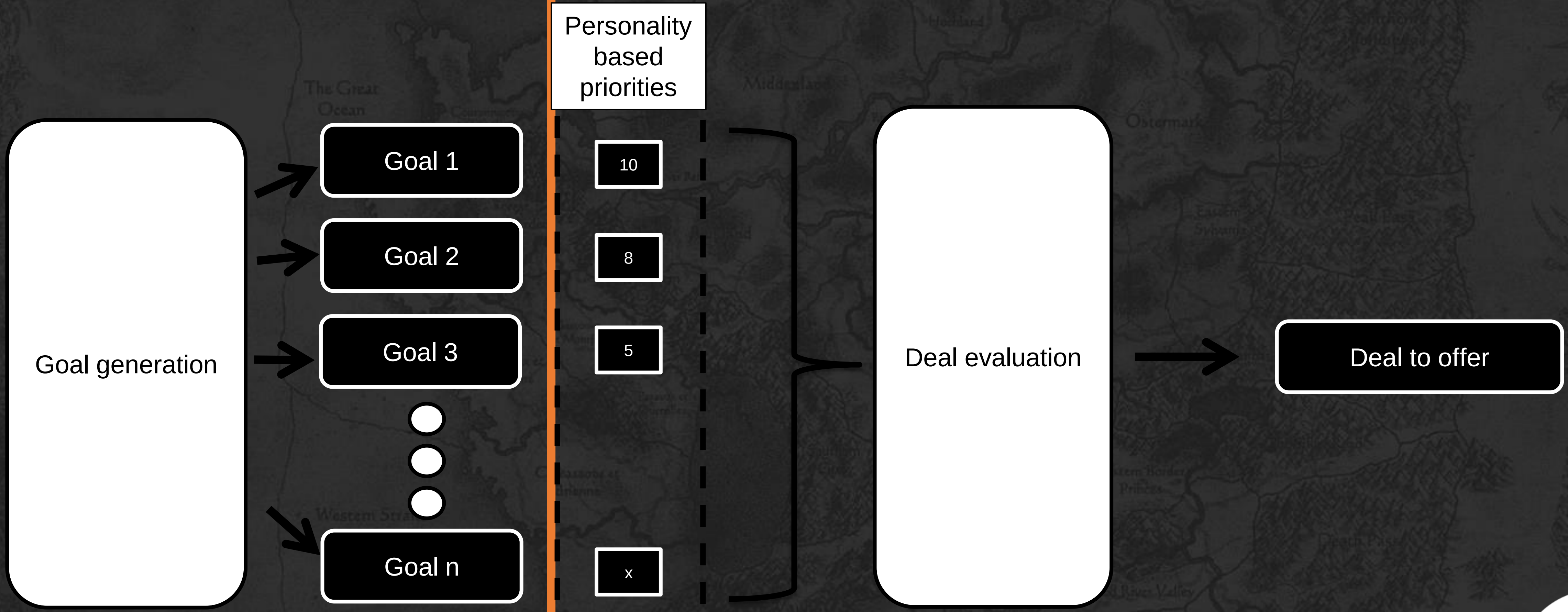
@aphrael

Diplomacy system



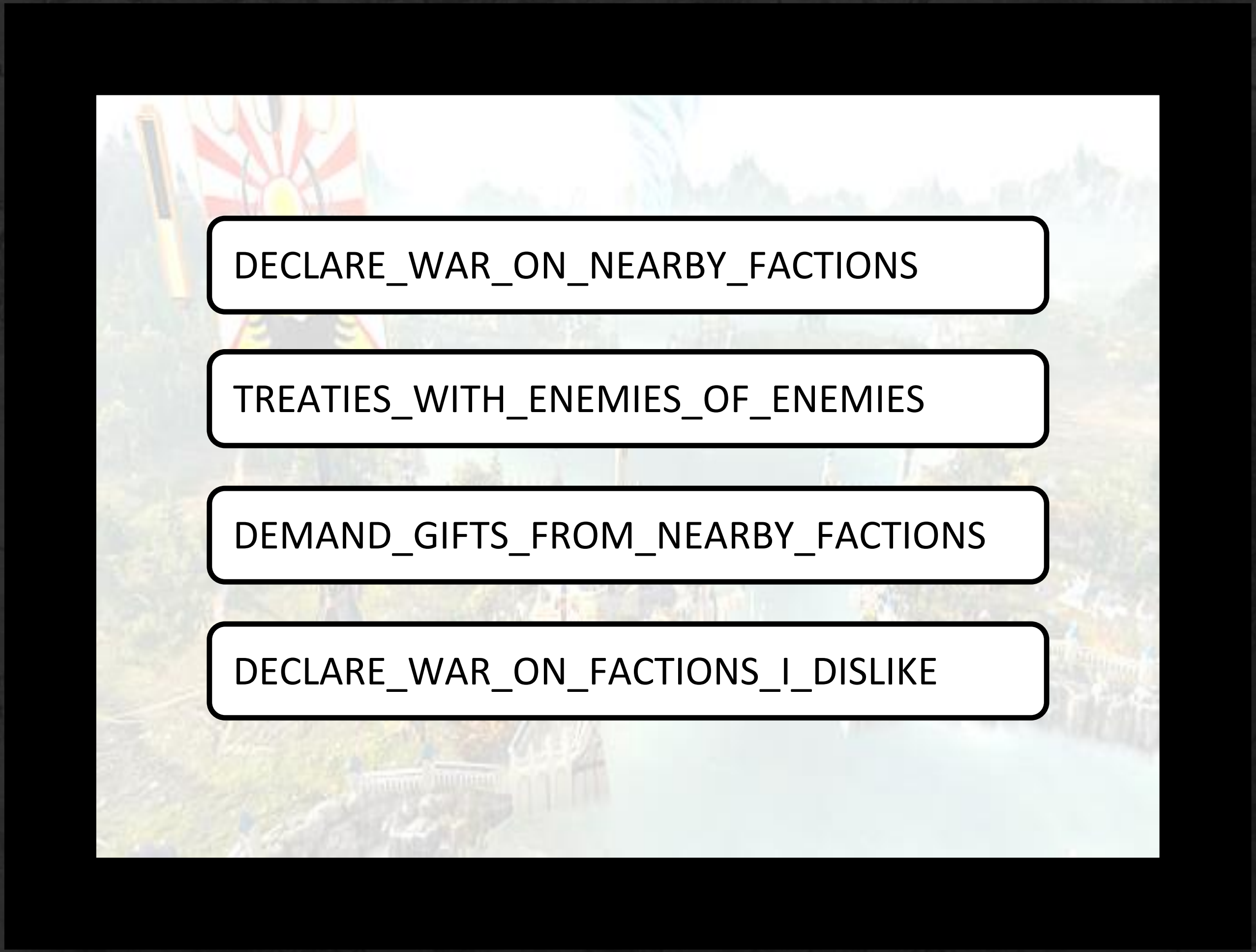
@aphrael

Goal Generation



@aphrael

Goal generation


DECLARE_WAR_ON_NEARBY_FACTIONS

TREATIES_WITH_ENEMIES_OF_ENEMIES

DEMAND_GIFTS_FROM_NEARBY_FACTIONS

DECLARE_WAR_ON_FACTIONS_I_DISLIKE

@aphrael

Goal generation

Goal Generators

DECLARE_WAR_ON_NEARBY_FACTIONS

TREATIES_WITH_ENEMIES_OF_ENEMIES

DEMAND_GIFTS_FROM_NEARBY_FACTIONS

DECLARE_WAR_ON_FACTIONS_I_DISLIKE

List of goals

- War to my bitter enemy Faction1
- War to unfriendly Faction2

- **Non aggression pact with Faction3**
- **Non aggression pact and demand money from Faction 1**
- **Trade agreement with Faction3**
- **Trade agreement with Faction1**

- Demand 1000 payment from neighbor Faction1
- Demand 200 regular payment from Faction2

- Declare war on Faction 1
- Declare war on Faction 2

@aphrael

Goal generation

```
// Peace
CAI_GOAL_GENERATORS::ASK_FOR_PEACE                    (balance_analyser, *this).generate_goals(goals, faction);

// War
CAI_GOAL_GENERATORS::DECLARE_WAR_ON_FACTIONS_I_DISLIKE(balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::OFFER_TO_JOIN_ALLYS_WAR          (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::REQUEST_ALLY_TO_JOIN_WAR         (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::DEMAND_VASSALAGE_OR_CLIENT_STATE (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::OFFER_VASSALAGE_TO_NEIGHBOURS    (balance_analyser, *this).generate_goals(goals, faction);

// Trade
CAI_GOAL_GENERATORS::OBTAIN_TRADE_AGREEMENT           (balance_analyser, *this).generate_goals(goals, faction);

// Gifts
CAI_GOAL_GENERATORS::DEMAND_GIFTS_FROM_WEAKER_FACTIONS(balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::OFFER_GIFTS_TO_STRONGER_FACTIONS (balance_analyser, *this).generate_goals(goals, faction);

// Treaties
CAI_GOAL_GENERATORS::TREATIES_WITH_NEIGHBOURS         (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::TREATIES_WITH_ALLIES_OF_ALLIES   (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::UPGRADE_TREATIES                 (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::MARRIAGE                         (balance_analyser, *this).generate_goals(goals, faction);
```

Goal generation

```
// Peace
CAI_GOAL_GENERATORS::ASK_FOR_PEACE                    (balance_analyser, *this).generate_goals(goals, faction);

// War
CAI_GOAL_GENERATORS::DECLARE_WAR_ON_FACTIONS_I_DISLIKE(balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::OFFER_TO_JOIN_ALLYS_WAR           (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::REQUEST_ALLY_TO_JOIN_WAR          (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::DEMAND_VASSALAGE_OR_CLIENT_STATE (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::OFFER_VASSALAGE_TO_NEIGHBOURS     (balance_analyser, *this).generate_goals(goals, faction);

// Trade
CAI_GOAL_GENERATORS::OBTAIN_TRADE_AGREEMENT            (balance_analyser, *this).generate_goals(goals, faction);

// Gifts
CAI_GOAL_GENERATORS::DEMAND_GIFTS_FROM_WEAKER_FACTIONS(balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::OFFER_GIFTS_TO_STRONGER_FACTIONS (balance_analyser, *this).generate_goals(goals, faction);

// Treaties
CAI_GOAL_GENERATORS::TREATIES_WITH_NEIGHBOURS          (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::TREATIES_WITH_ALLIES_OF_ALLIES    (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::UPGRADE_TREATIES                  (balance_analyser, *this).generate_goals(goals, faction);
CAI_GOAL_GENERATORS::MARRIAGE                          (balance_analyser, *this).generate_goals(goals, faction);
```

Goal generation

```
for (const CAI_PERSONALITY_DEAL_GENERATION_GENERATOR_RECORD* generator : deal_generators())
{
        CAI_DIPLOMATIC_GOAL_GENERATOR(*generator).generate_goals(goals, nullptr);
}
```

@aphrael

Goal generation

```
for (const CAI_PERSONALITY_DEAL_GENERATION_GENERATOR_RECORD* generator : deal_generators())
{
		CAI_DIPLOMATIC_GOAL_GENERATOR(*generator).generate_goals(goals, nullptr);
}
```

@aphrael

Goal generators

Generator Key	Goal key	Target group	Condition set
BREAK_TREATIES	**goal_break_military_access**	**cai_target_group_known_factions**	**BREAK_MILITARY_ACCESS_CONDITION_SET**
BREAK_TREATIES	goal_break_non_aggression	cai_target_group_known_factions	BREAK_NON_AGRESSION_CONDITION_SET
BREAK_TREATIES	goal_break_trade	cai_target_group_known_factions	BREAK_TRADE_CONDITION_SET
BREAK_TREATIES	goal_break_offer_regular_payments	cai_target_group_known_factions	BREAK_OFFER_REGULAR_PAYMENTS_CONDITION_SET
TREATIES_WITH_NEARBY_NONHOSTILES	goal_non_aggression_pact	cai_target_group_neighbours	NON_AGRESSION_PACT_CONDITION_SET
KICK_DISLIKED_ALLIANCE_MEMBER	goal_kick_coalition_member	cai_target_group_allies	KICK_COALITION_MEMBER_CONDITION_SET
DEMAND_ANCILLARY_FROM_FRIEND	goal_demand_ancillary	cai_target_group_known_factions	ANCILLARY_CONDITION_SET

Goal generation: Treaties

Goal key	Mandatory treaty key
goal_break_military_access	treaty_components_break_military_access
goal_break_non_aggression	treaty_components_break_non_aggression
goal_break_offer_food	treaty_components_break_food_supply_offer
goal_break_offer_regular_payments	treaty_components_break_payment_regular_offer
goal_break_trade	treaty_components_break_trade
goal_demand_ancillary	treaty_components_ancillary_demand
goal_demand_food	treaty_components_food_supply_demand

Goal key	Optional treaty key	Priority
goal_break_military_access	treaty_components_ancillary_demand	2
goal_break_military_access	treaty_components_payment_offer	1

@aphrael

Goal generation: Conditions

Condition_set_key	Condition_key	Evaluates_to	Param_faction	owner	status	treaty	stance	value
BREAK_MILITARY_ACCESS_CONDITION_SET	deal_generation_condition_strategic_stance_better_than	false	respondent_recipient	respondent_proposer			strategic_stance_friendly	0
OFFER_VASSALAGE_CONDITION_SET	deal_generation_condition_target_faction_is_stronger_than_me_by_at_least	true	respondent_recipient	respondent_proposer				0.2
OFFER_VASSALAGE_CONDITION_SET	deal_generation_condition_has_treaty	false				treaty_components_vassalage		

@aphrael

Goal generation: Conditions

```
class CAI_DIPLOMATIC_GOAL_GENERATION_CONDITION_FUNCTIONS :
           public CAI_DEAL_COMPONENT_EVALUATION_FUNCTIONS<CAI_CONDITION_RECORD, GEN_COND_FUNC>
{
public:

CAI_DIPLOMATIC_GOAL_GENERATION_CONDITION_FUNCTIONS();

private:

void                      init_map                    () override;
static CONDITION_RESPONSE has_treaty                  (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
static CONDITION_RESPONSE has_treaty_with_anyone      (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
static CONDITION_RESPONSE in_alliance                 (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
static CONDITION_RESPONSE knows                       (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
static CONDITION_RESPONSE neutral_with                (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
static CONDITION_RESPONSE strategic_stance_better_than(const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
static CONDITION_RESPONSE target_faction_is_stronger_than_me_by_at_least
                                                      (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to);
};
```

@aphrael

Goal generation: Conditions

```
CONDITION_RESPONSE
CAI_DIPLOMATIC_GOAL_GENERATION_CONDITION_FUNCTIONS::target_faction_is_stronger_than_me_by_at_least
    (const CAI_FACTION& owner_faction, const PARAMETERS& parameters, bool evaluates_to)
{
    if (parameters.param_faction == nullptr)
    {
        return CONDITION_RESPONSE_IRRELEVANT;
    }

    const CAI_FACTION_ALLIANCE_STRENGTH_ANALYSIS& strength_analysis_us =
         faction_alliance_strength_analyser().determine_analysis_for(deconst(owner_faction));
    const CAI_FACTION_ALLIANCE_STRENGTH_ANALYSIS& strength_analysis_them =
         faction_alliance_strength_analyser().determine_analysis_for(deconst(*param_faction));

    if (strength_analysis_us.strength() != 0 && (strength_analysis_them.strength() -
         strength_analysis_us.strength() / strength_analysis_us.strength() > param_value) == evaluates_to)
    {
         return CONDITION_RESPONSE_HOLDS;
    }
    return CONDITION_RESPONSE_DOES_NOT_HOLD;
}
```

@aphrael

Goal generation: Target groups

```cpp
class CAI_DIPLOMATIC_GOAL_GENERATION_TARGET_GROUP_FUNCTIONS :
             public CAI_DEAL_COMPONENT_EVALUATION_FUNCTIONS<CAI_TARGET_GROUP_RECORD, TARGET_FUNC>
{
public:
    CAI_DIPLOMATIC_GOAL_GENERATION_TARGET_GROUP_FUNCTIONS();
private:
    void        init_map              () override;
    static void allies                (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void allies_of_allies      (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void disliked_factions     (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void enemies               (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void factions_with_treaties(CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void known_factions        (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void neighbours            (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
    static void horde_neighbours      (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions);
};
```

Goal generation: Target groups

```
void CAI_DIPLOMATIC_GOAL_GENERATION_TARGET_GROUP_FUNCTIONS::neighbours
    (CAI_BDI_POOL& bdi_pool, CAI_FACTION& faction, CA_STD::VECTOR<const CAI_FACTION*>& factions)
{
    CAI_NEIGHBOURING_FACTIONS_ANALYSIS& neighbour_analysis =
         bdi_pool.get_central_bdi_pool().neighbouring_factions_analyser().determine_analysis_for(faction);

   factions.insert(factions.end(), neighbour_analysis.neighbours_begin(), neighbour_analysis.neighbours_end());
}
```

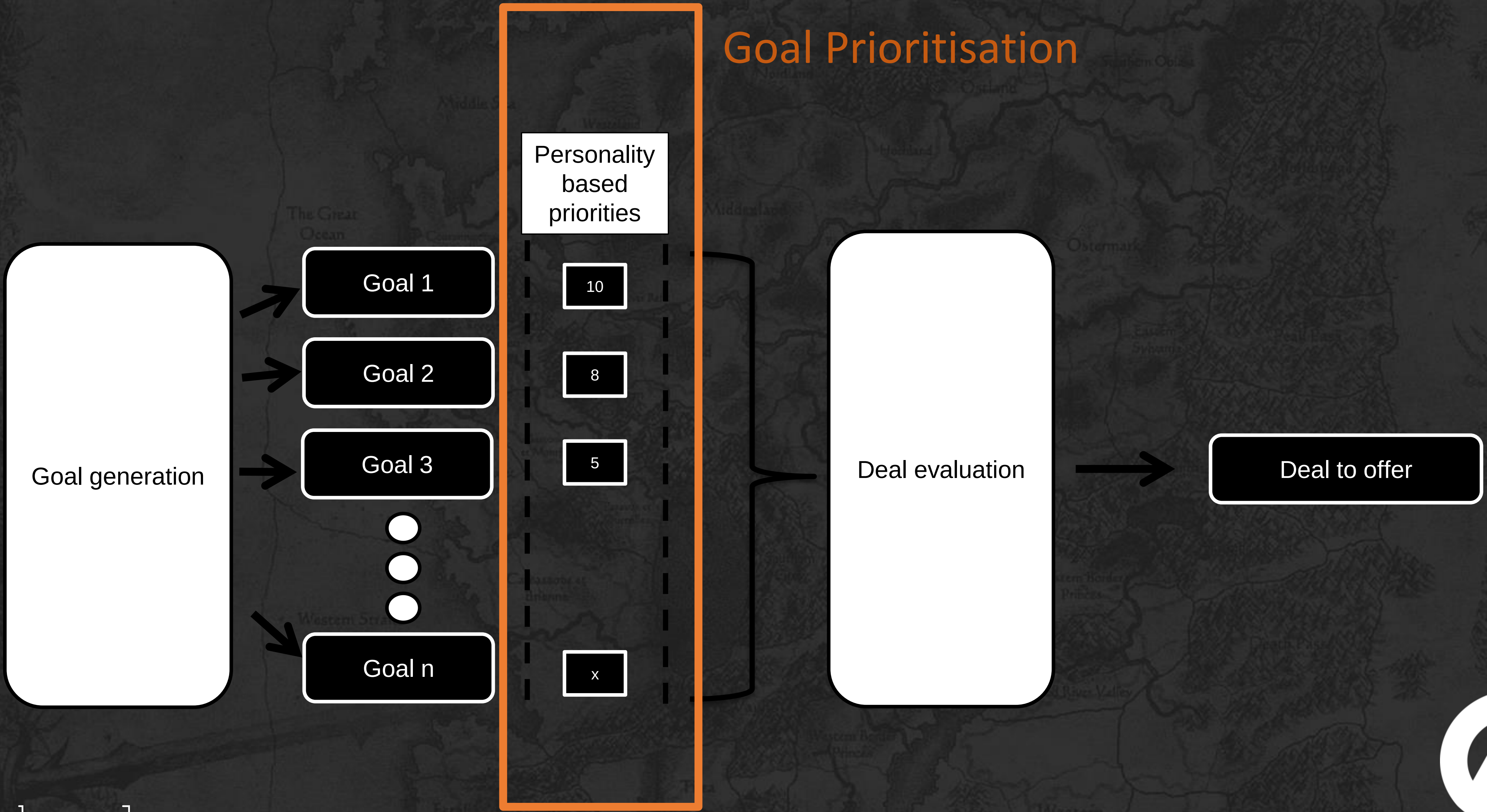
@aphrael

CAI_DIPLOMATIC_GOAL_GENERATOR::generate_goals

```
void CAI_DIPLOMATIC_GOAL_GENERATOR::generate_goals(CA_STD::VECTOR<CAI_DIPLOMATIC_GOAL>& goals)
{
    const CAI_DIPLOMATIC_GOAL_GENERATION_GOAL_TEMPLATES& deal_generation_goals =
           cai_diplomatic_goal_bases().deal_generation_goals(*m_generator_record);
    bool conditions_satisfied = true;
    if ((*goal_template).m_condition_data_vector != nullptr)
    {
       for (const CAI_FACTION* target : target_factions)
       {
          for (CAI_DIPLOMATIC_GOAL_GENERATION_CONDITION_DATA condition : (*goal_template).m_condition_data_vector)
          {
              PARAMETERS params(&(*cond_itr).m_param_status->m_faction_status, (*cond_itr).m_param_treaty,
(*cond_itr).m_param_stance,                                    (*cond_itr).m_param_value);
              CONDITION_RESPONSE response = (*cond_itr).m_func(*owner_faction, nullptr, params, (*cond_itr).m_evaluates_to);
              if (response == CONDITION_RESPONSE_DOES_NOT_HOLD)
              {
                 conditions_satisfied = false;
                 break;
              }
           }
       }
       if (conditions_satisfied)
       {
          goals.emplace_back(*(*goal_template).m_goal, m_faction, *target, values, m_failure_timeout, m_priority_base);
       }
    }
}
```

@aphrael

Goal Prioritisation



@aphrael

Goal prioritisation

List of goals

- War to my bitter enemy Faction1
- War to unfriendly Faction2

- Non aggression pact with Faction3
- Non aggression pact & demand money from Faction 1
- Trade agreement with Faction3
- Trade agreement with Faction1

- Demand 1000 payment from neighbor Faction1
- Demand 200 regular payment from Faction2

- Make peace, and become my vassal to Faction1
- Make peace, and become my vassal to Faction2

Prioritisation

- **(20)** War to my bitter enemy Faction1
- **(10)** War to unfriendly Faction2

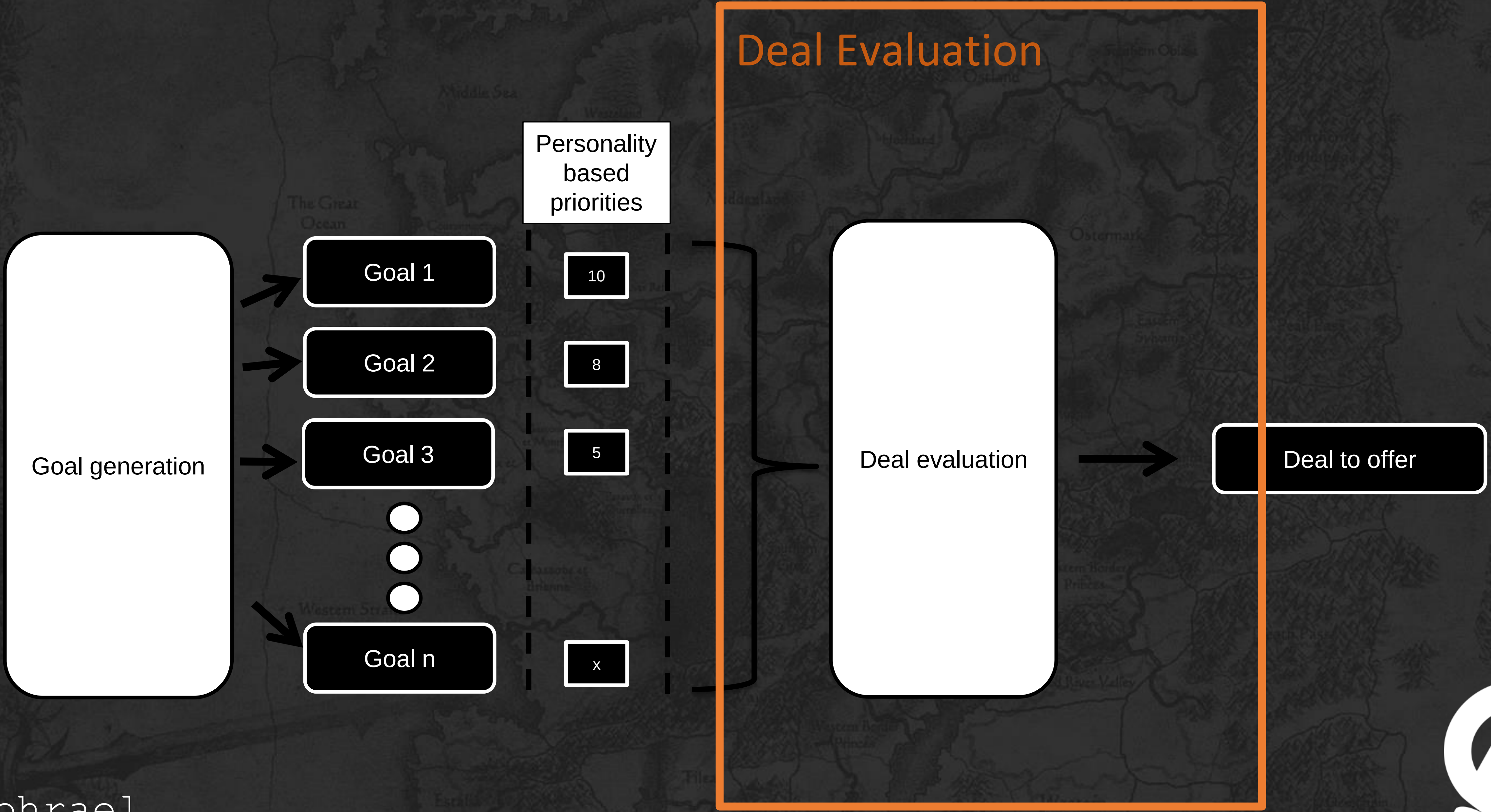
- **(5)** Non aggression pact with Faction3
- **(2)** Non aggression pact & demand money from Faction 1
- **(10)** Trade agreement with Faction3
- **(0)** Trade agreement with Faction1

- **(8)** Demand 1000 payment from neighbor Faction1
- **(6)** Demand 200 regular payment from Faction2

- **(1)** Make peace, and become my vassal to Faction1
- **(3)** Make peace, and become my vassal to Faction2

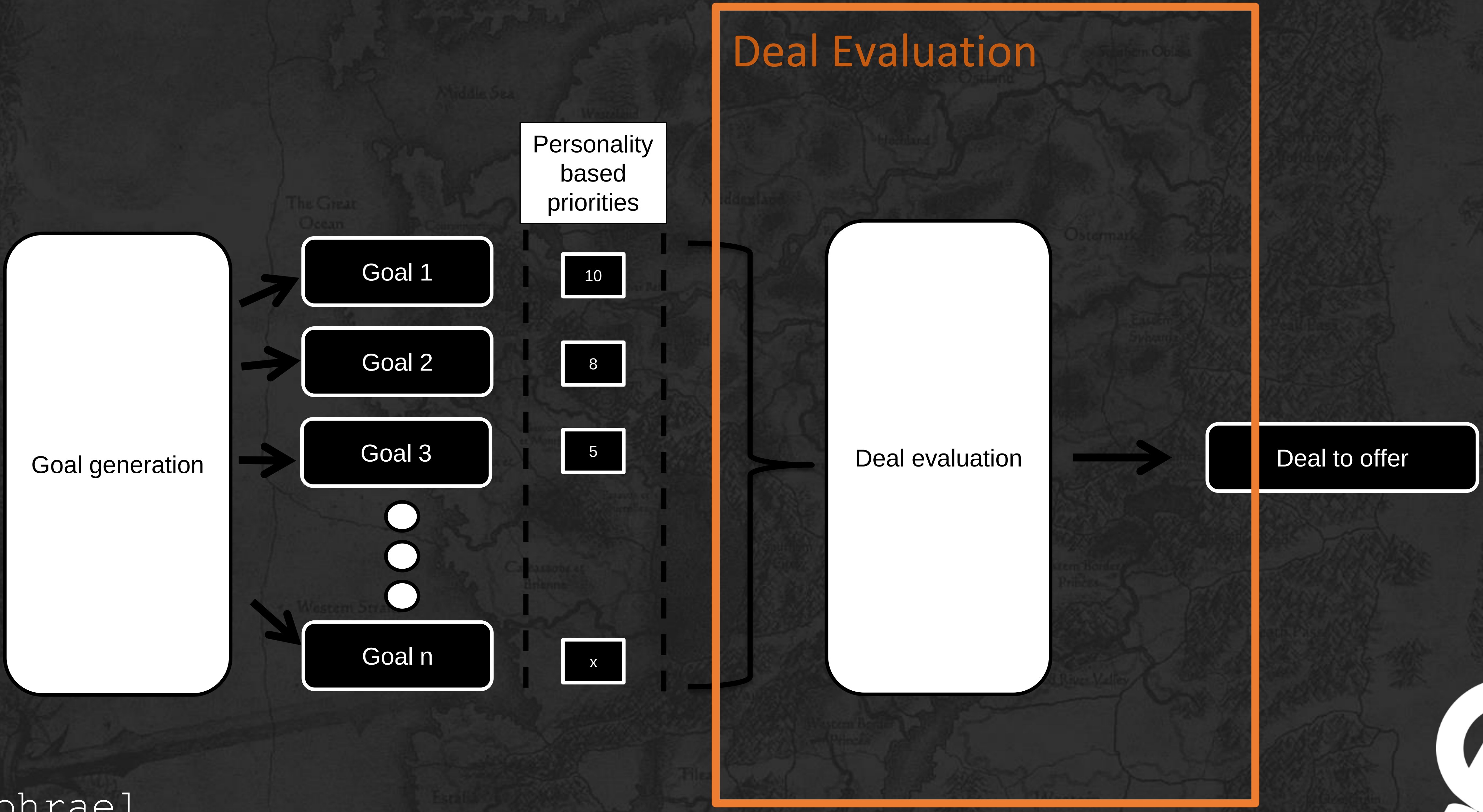
@aphrael

Deal Evaluation



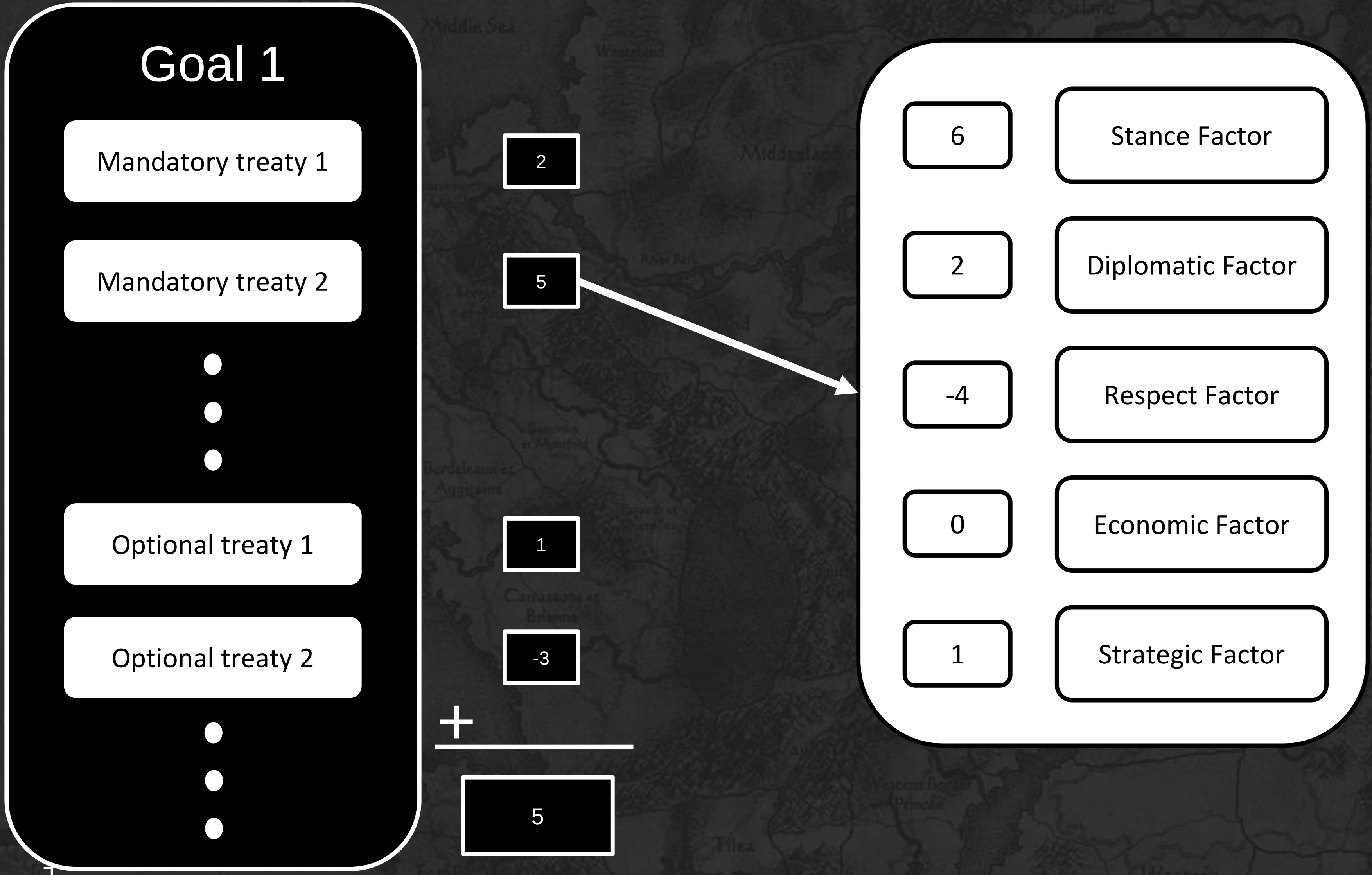
@aphrael

Deal Evaluation



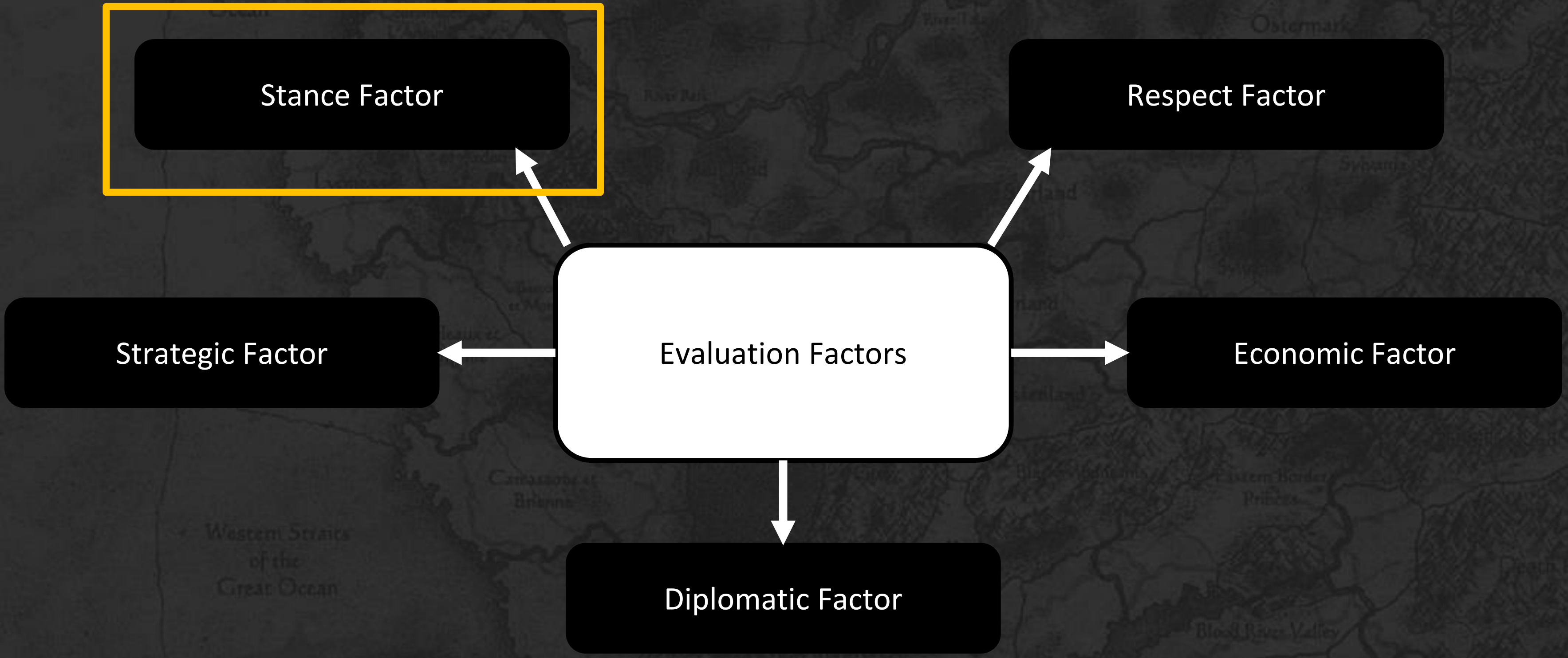
@aphrael

Deal evaluation



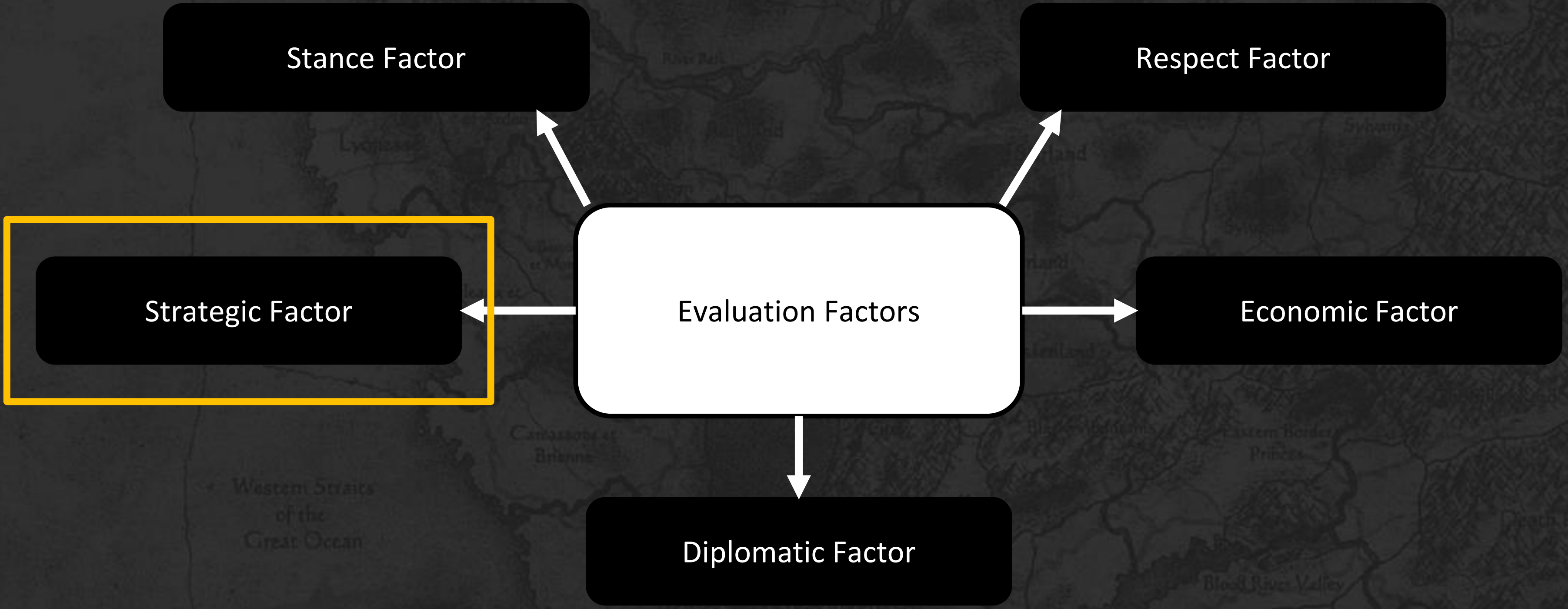
@aphrael

Evaluation factors



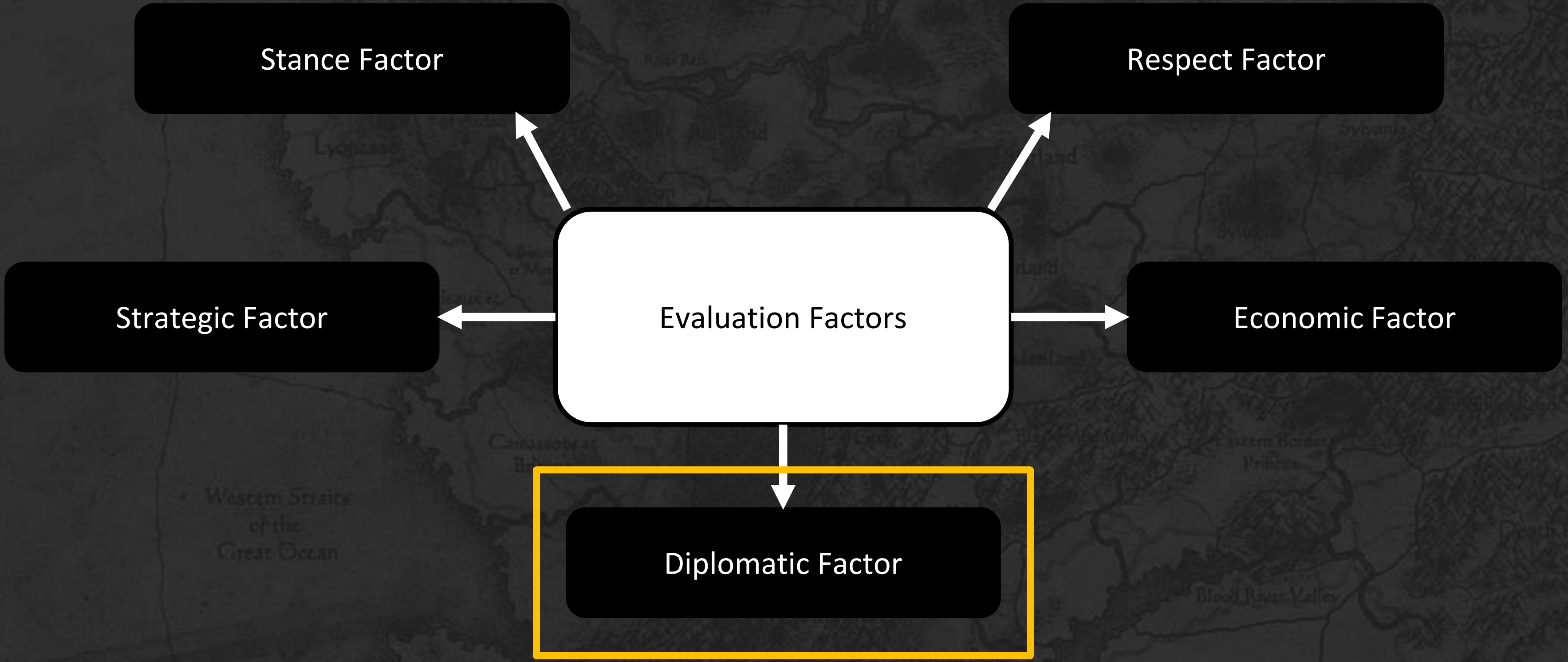
@aphrael

Evaluation factors



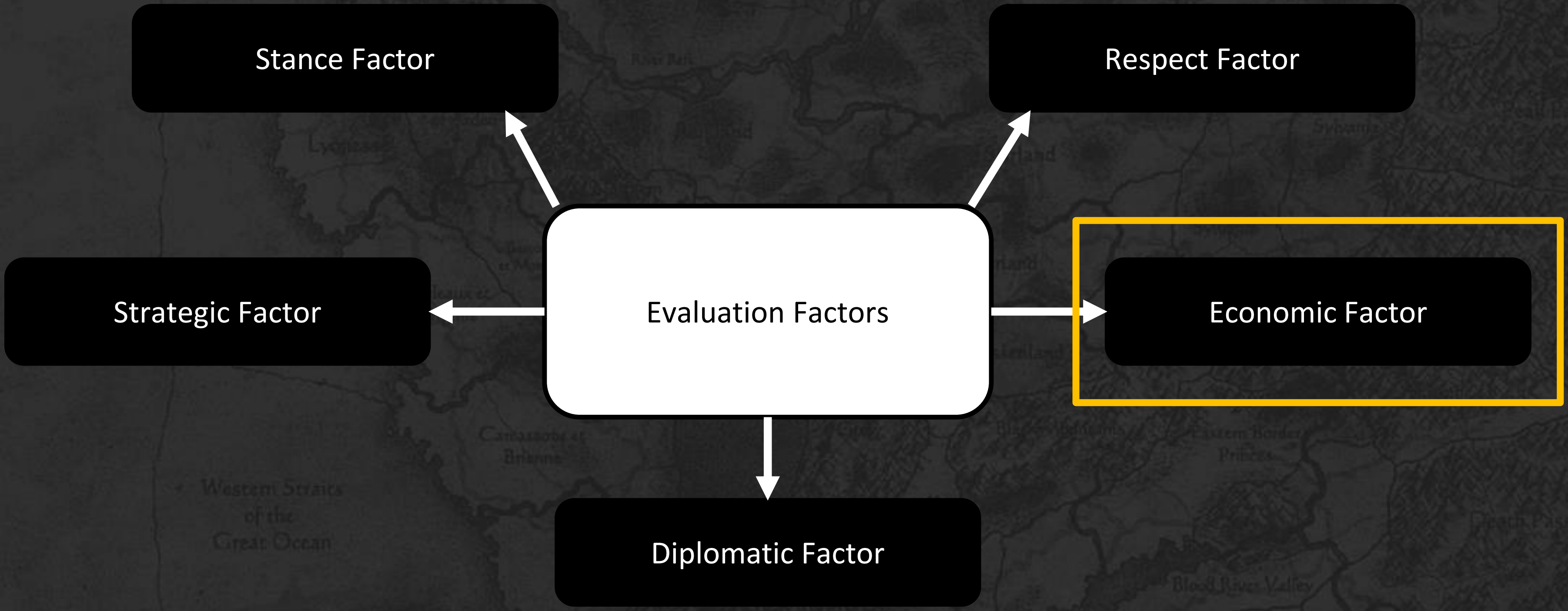
@aphrael

Evaluation factors



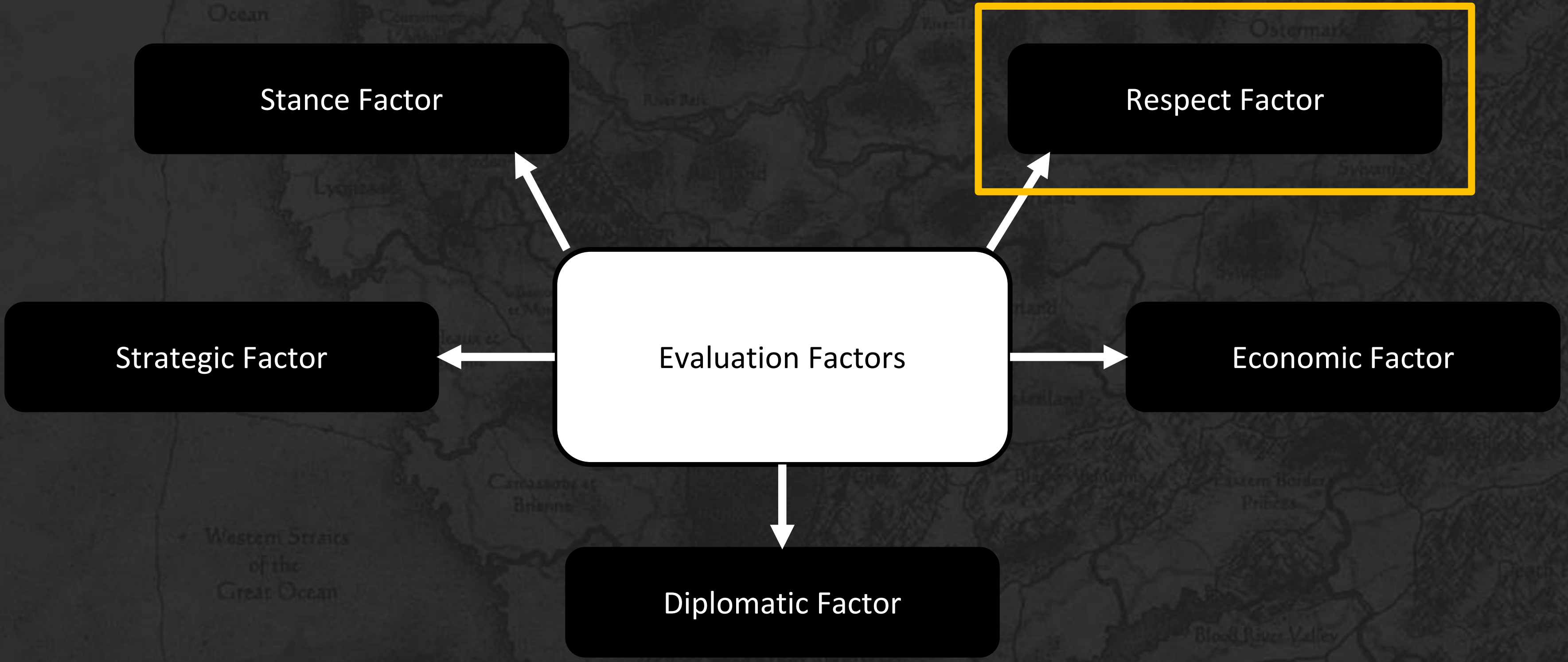
@aphrael

Evaluation factors



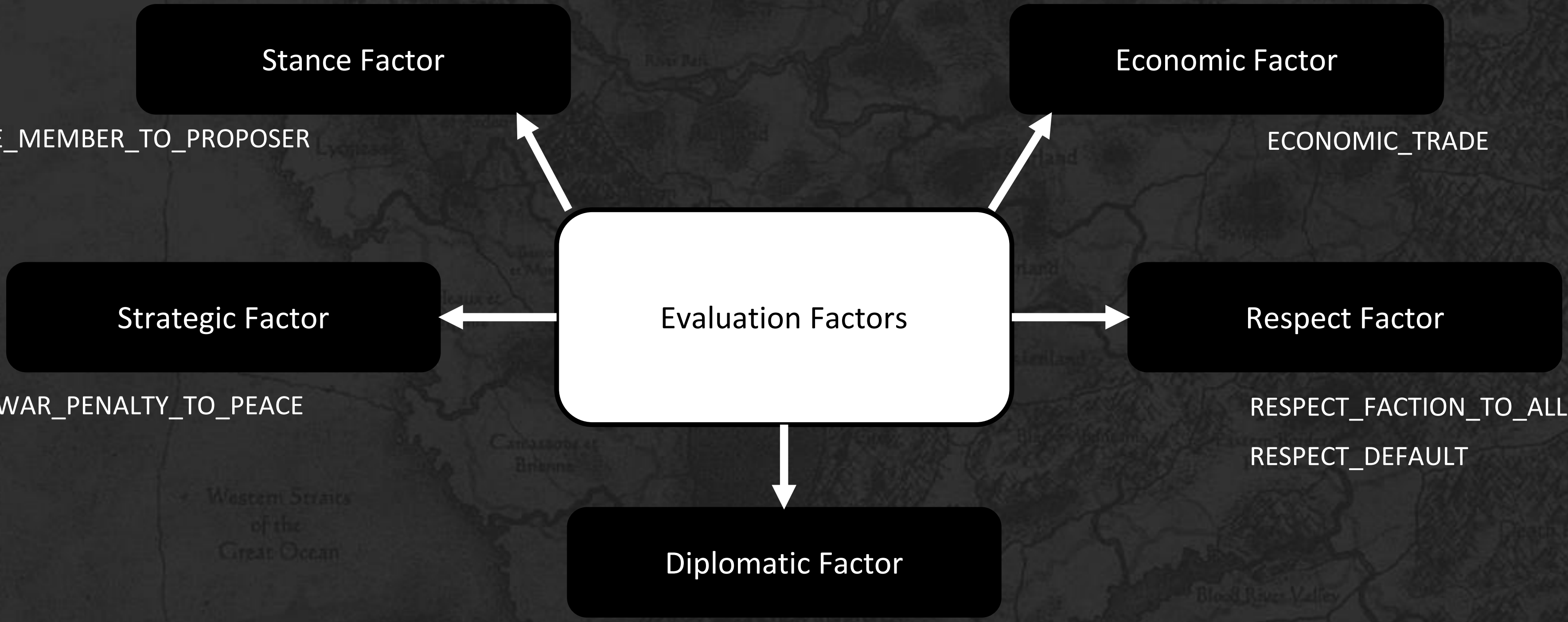
@aphrael

Evaluation factors



@aphrael

Evaluation factors: Criteria



STANCE_ALLIANCE_MEMBER_TO_PROPOSER

ECONOMIC_TRADE

STRATEGIC_RECENT_WAR_PENALTY_TO_PEACE

RESPECT_FACTION_TO_ALLIANCE_MEMBERS

RESPECT_DEFAULT

DIPLOMACY_CONSEQUENCES_OF_BREAK_ALL_TREATY_EVENTS

DIPLOMACY_CONSEQUENCES_OF_JOIN_FACTIONS_WARS

@aphrael

Evaluation factors

Factor	Criterion type
diplomatic_factor	DIPLOMACY_CONSEQUENCES_OF_EVENT
strategic_factor	STRATEGIC_STRENGTH_OUR_AND_VASSALS_TO_TOTAL_IF_POSITIVE_BALANCE
strategic_factor	STRATEGIC_ALLIANCE_SIZE_PENALTY
respect_factor	RESPECT_DEFAULT
economic_factor	ECONOMIC_ZERO

Treaty component	Criterion type	negate
break_non_aggression	STANCE_DEFAULT	false
break_non_aggression	STRATEGIC_STRENGTH_OUR_AND_VASSALS_TO_TOTAL_IF_POSITIVE_BALANCE	true
break_non_aggression	DIPLOMACY_CONSEQUENCES_OF_EVENT	false
break_non_aggression	**ECONOMIC_ZERO**	**false**
break_non_aggression	RESPECT_DEFAULT	false

@aphrael

Evaluation factors: Criteria functions

```
class CAI_DEAL_COMPONENT_EVALUATION_CRITERIA_FUNCTIONS :
    public CAI_DEAL_COMPONENT_EVALUATION_FUNCTIONS<CAI_DIPLOMACY_DEAL_EVALUATION_CRITERION_TYPE_RECORD, EVAL_FUNC>
{

public:
    CAI_DEAL_COMPONENT_EVALUATION_CRITERIA_FUNCTIONS();

private:
    void           init_map() override;

    static float32 character_default_demand           (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 consequences_of_join_alliances_wars(TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 economic_regular_demand            (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 economic_regular_offer             (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 consequences_of_break_all_treaties (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 respect_default                    (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 respect_faction_to_alliance_members(TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 stance_faction_to_alliance_members (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 strategic_alliance_size_penalty    (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 strategic_recent_peace_penalty     (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
    static float32 strategic_victory_region           (TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params);
};
```

@aphrael

Evaluation factors: Criteria functions

```
float32 CAI_DEAL_COMPONENT_EVALUATION_CRITERIA_FUNCTIONS::
consequences_of_break_all_treaties(TREATY_COMPONENT_RECORD& record, COMPONENT_DATA& component_data, PARAMS params)
{
    const FACTION& evaluator_faction = component_data.m_evaluator().get_campaign_faction();
    const FACTION& opponent_faction = component_data.m_opponent()->get_campaign_faction();
    const DIPLOMACY::DEAL::DEALS & deals = evaluator_faction.campaign_model().diplomacy().active_deals();
    CA_STD::VECTOR<CAI_LOG_MANAGER::SPECULATIVE_COMPONENT> components;
    for (const DIPLOMACY::DEAL& deal : deals)
    {
        if (DIPLOMACY::faction_involved_in_deal(evaluator_faction, deal) &&
            DIPLOMACY::faction_involved_in_deal(opponent_faction, deal))
        {
            for (const DIPLOMACY::DEAL_COMPONENT& deal_component : deal.components())
            {
                components.push_back(CAI_LOG_MANAGER::SPECULATIVE_COMPONENT(evaluator_faction, opponent_faction,
                    &deal_component.component(), DEAL_COMPONENT_CHANGE_TYPE::BROKEN));
            }
        }
    }
    return diplomatic_value(bdi_pool, components,component_data.m_evaluator(), proposer);
}
```

@aphrael

Deal evaluation - value of a treaty component

```
float32 CAI_DEAL_EVALUATION::component_value(CAI_BDI_POOL& bdi_pool,
    const DIPLOMACY::NEGOTIATION_STATE& negotiation_state, CAI_FACTION& evaluator, CAI_FACTION& to,
    CAI_FACTION &proposer, const CAMPAIGN_DIPLOMACY_TREATY_COMPONENT_RECORD& record, int32 value)
{
    CAI_DEAL_COMPONENT deal_component(bdi_pool.get_central_bdi_pool(), negotiation_state,
        evaluator, to, proposer, record, value);

    float32 total_component_value = 0.0f;
    const CAI_DIPLOMACY_DEAL_EVALUATION_FACTOR_TYPES_TABLE& factors = deal_evaluation_factor_types_table();
    for (const CAI_DIPLOMACY_DEAL_EVALUATION_FACTOR_TYPE_RECORD *factor : factors)
    {
        total_component_value += deal_component.deal_evaluation_factor(*factor);
    }

    return total_component_value;
}
```

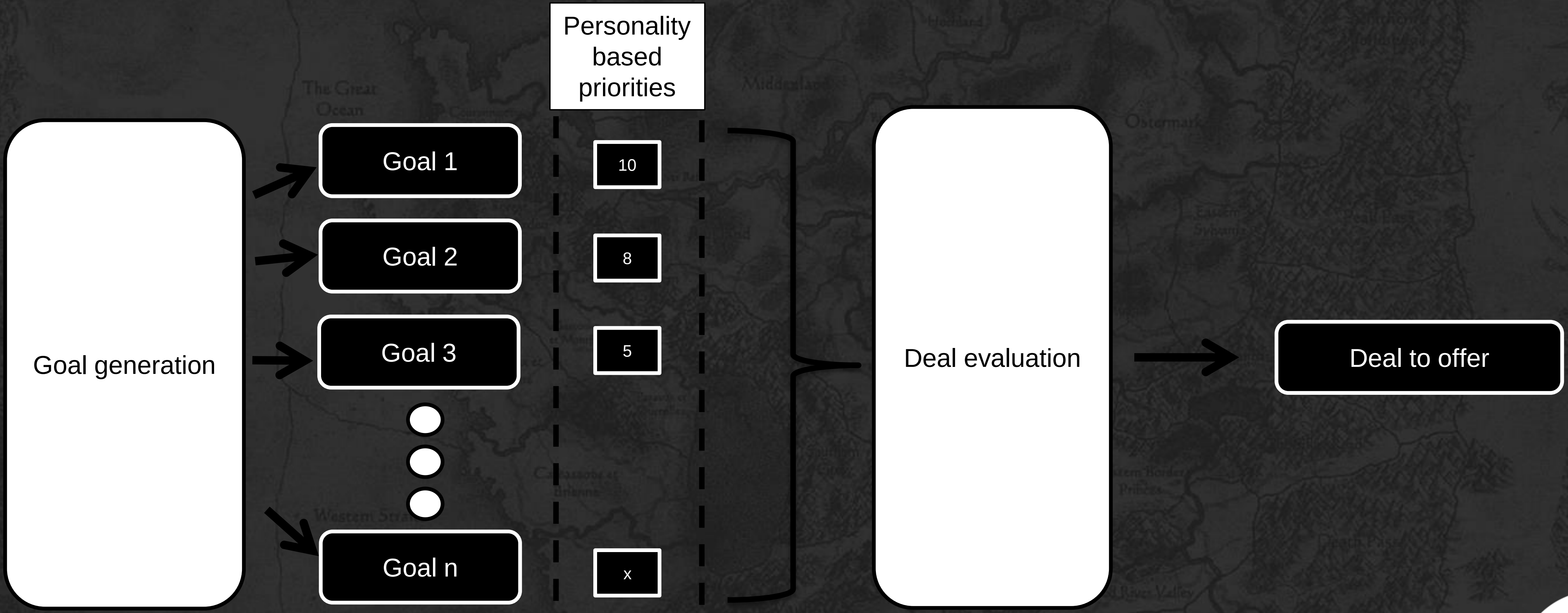
@aphrael

CAI_DEAL_COMPONENT::deal_evaluation_factor

```
float32 CAI_DEAL_COMPONENT::deal_evaluation_factor(const DEAL_EVALUATION_FACTOR_TYPE_RECORD& factor)
{
    DEAL_CRITERIA_MAP::const_iterator criteria_map_itr = m_deal_evaluation_criterias_by_factor->find(&factor);

    float32 sum = 0.0f;
    for (const CRITERION_BASE& criteria_base : criteria_map_itr->second)
    {
        COND_FUNC condition = cai_deal_component_evaluation_conditions.func(criteria_base.m_condition);

        if (condition == nullptr ||
           condition(m_bdi_pool, m_component_data.m_evaluator, m_recipient(), &m_proposer(), m_params))
        {
            EVAL_FUNC eval = cai_deal_component_evaluation_criteria.func(criteria_base.m_criterion_type);

            sum += eval(m_bdi_pool, *m_record(), m_component_data, m_proposer(), m_params) *
                          (criteria_base.m_should_negate ? -1.0f : 1.0f);
        }
    }
    return sum;
}
```

@aphrael

Diplomacy system

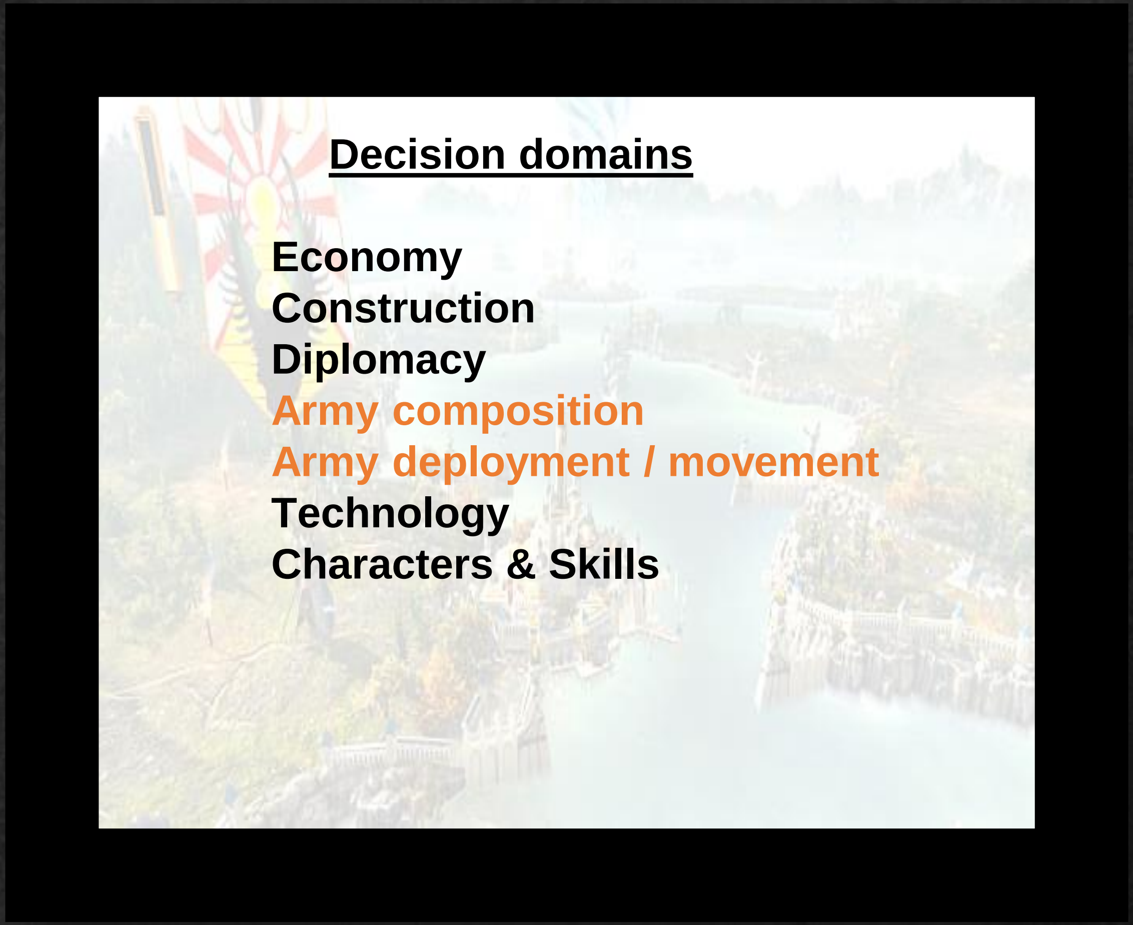


@aphrael

Overview

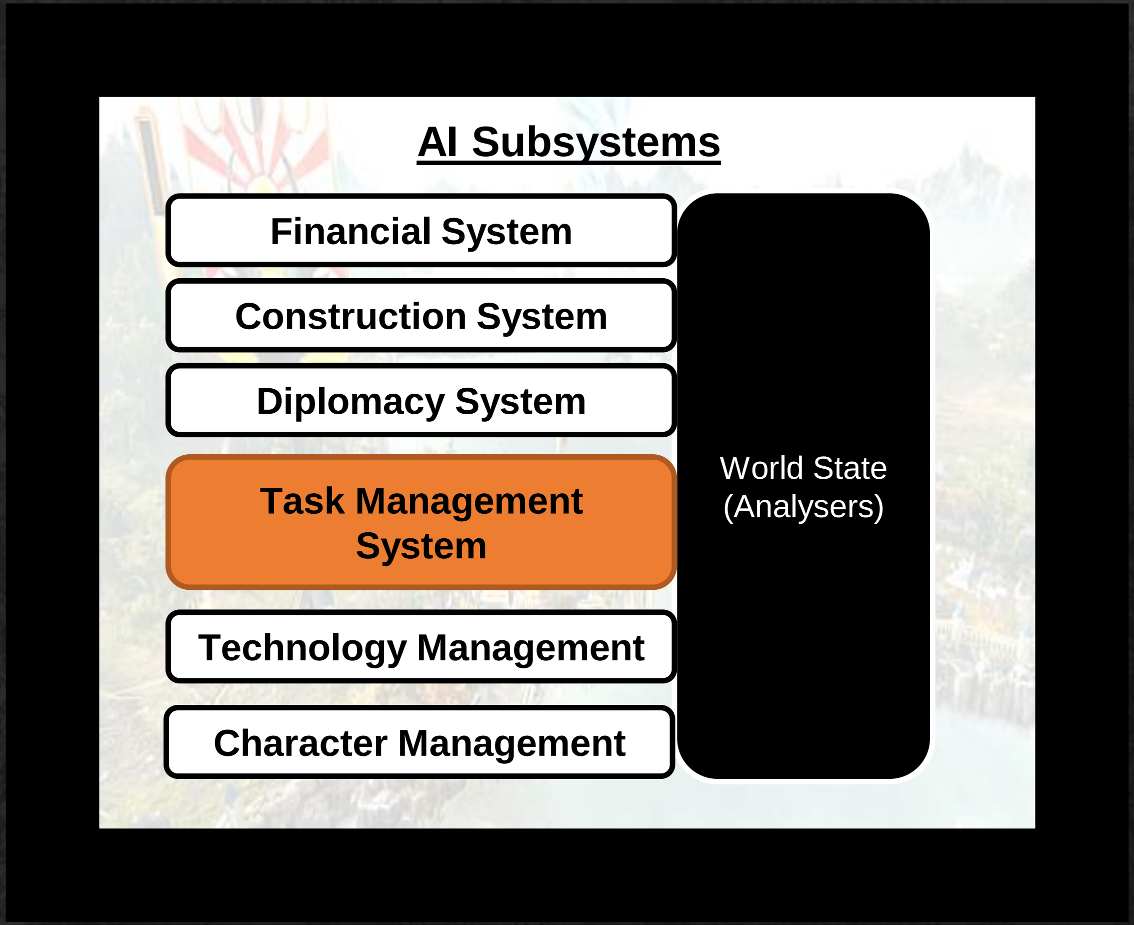
- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy
- Tasks and resources
- Profiling and timing

@aphrael



Decision domains

Economy
Construction
Diplomacy
Army composition
Army deployment / movement
Technology
Characters & Skills



AI Subsystems

@aphrael

Task management system

@aphrael

Task management system

Task Generation

Resource Allocation

Resource Coordination

@aphrael

Task management system

Task Generation

Resource Allocation

Resource Coordination

@aphrael

Task generation

Attack region

Attack force

Recruit new army

Raid region

Defend region

Embed agent

Recruit in region

@aphrael

Task generation: attack region

```
// =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-= CAI_TASK_ATTACK_REGION =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
// Task for performing simple region attack
class CAI_TASK_ATTACK_REGION : public CAI_TASK
{
public:
CAI_TASK_ATTACK_REGION(CAI_REGION &region, CAI_TASK_RESOURCE_REALLOCATION_BASE_POLICY base_resource_realloc_policy);

virtual void                        on_pool_add                         (CAI_BDI_POOL &bdi_pool);

// Resource Allocation
virtual CAI_TASK_REQUIREMENTS *     generate_requirements_information   (CAI_BDI_POOL &bdi_pool, const
                                                                         CAI_STANCE_INFORMATION &stance_info);
virtual CAI_TASK_TARGET_OBJECTIVE   generate_requirements_objective     () const;
virtual void                        populate_stance_information         (CAI_BDI_POOL &bdi_pool,
                                                                         CAI_STANCE_INFORMATION_VECTOR &stance_info);
virtual CAI_FACTION *               primary_task_target_faction         ();
virtual const CAI_FACTION *         primary_task_target_faction         () const;

private:
// Operation
virtual void                        execute                             (CAI_BDI_POOL &bdi_pool);

CONST_SAFE_PTR<CAI_REGION>           m_region;
};
```

@aphrael

Task generation: generate requirements 1/2

```
CAI_TASK_REQUIREMENTS* CAI_TASK_ATTACK_REGION::generate_requirements_information
    (CAI_BDI_POOL &bdi_pool, const CAI_STANCE_INFORMATION &/*stance_info*/)
{
    CAI_TASK_REQUIREMENT_TARGET target_info = generate_requirments_target();
    CAI_TASK_REQUIREMENT_BOUNDS army_strength_bounds(0, 0, 0);
    CAI_TASK_REQUIREMENT_BOUNDS time_bounds(0, 0, 0);

    CAI_FACTION &our_faction = bdi_pool.get_faction_bdi_pool()->get_faction();
    CAI_RESOURCE_MOBILE_MILITARY_STRENGTH required_army_strength;
    CAI_MILITARY_STRENGTH_ANALYSER &military_strength_analyser = bdi_pool.military_strength_analyser();

    for (auto mobile : m_region->mobiles_in_region())
    {
       if (mobile->is_an_army())
       {
          required_army_strength += military_strength_analyser.determine_analysis_for(*mobile).absolute_strength();
       }
    }

    if( m_region->has_settlement() )
    {
        const CAI_GARRISONABLE_MILITARY_STRENGTH_ANALYSIS& region_strength_analysis =
                                military_strength_analyser.determine_analysis_for(*m_region->get_settlement());
        required_army_strength += region_strength_analysis.citizenry_strength_land();
    }
```

@aphrael

Task generation: generate requirements 2/2

```
card32 recommended_minimum = ca_round_to_card(static_cast<float32>(recommended_minimum) *
our_faction.faction_personality().strategic_component().enemy_strength_modifier());

    army_strength_bounds.m_minimum = recommended_minimum;
    army_strength_bounds.m_recommended = recommended_minimum * 2;
    army_strength_bounds.m_maximum = recommended_minimum * 8;

// Attritional Effects
    if( m_region->has_settlement() )
    {
        if(!our_faction.is_immune_to_attrition(*attrition_record) )
        {
            float32 attrition_multiplier        = m_region->get_settlement()->is_fortified() ? 2.5f : 1.5f;
            army_strength_bounds.m_minimum     *= attrition_multiplier;
            army_strength_bounds.m_recommended *= attrition_multiplier;
            army_strength_bounds.m_maximum     *= attrition_multiplier;
        }
     }
    return new CAI_TASK_REQUIREMENTS(*this, target_info, army_strength_bounds, time_bounds,
                   CAI_ALLOCATION_ACCEPTANCE_PREFERENCE_AT_LEAST_MINIMUMS, CAI_ARMY_PREFERENCE_ALWAYS_REQUIRED,
                   CAI_AGENT_PREFERENCE_AT_LEAST_ONE_IF_POSSIBLE, CAI_TIMING_PREFERENCE_ENFORCE,
                  CAI_FORCE_RECRUITMENT_REQUIREMENT::UNSPECIFIED,
                  CAI_FORCE_RECRUITMENT_REQUIREMENT::UNSPECIFIED );
}
```

@aphrael

Task generation: attack neighbouring regions

```
void generate_attack_neighbouring_war_regions_tasks(const CAI_ATTACK_REGION_TASK_SPECIFICATION::TASK_TYPE_MAP_NCC &,
CAI_ATTACK_REGION_TASK_SPECIFICATION::TASK_INSTANTIATION_INFO_PRIORITY_DATA_VECTOR &new_tasks_with_priority,
CAI_TASK_GENERATOR_CONTROL_DATA &task_generator_control_data, BDI_COMPONENT_ARRAY &)
{
    CAI_DIPLOMATIC_ANALYSIS &da_us = task_generator_control_data.diplomatic_analyser().determine_analysis_for(us);
    CA_STD::UNORDERED_MAP<card32, CAI_REGION *> regions_to_attack;
    std::for_each(task_generator_control_data.get_faction().regions_begin(),
         task_generator_control_data. get_faction().regions_end(),
                [&us, &da_us, &regions_to_attack](CAI_REGION *region)
    {
        std::for_each(region->neighbours_begin(), region->neighbours_end(), [& ](CAI_REGION_BOUNDARY *boundary)
        {
            CAI_REGION &other = boundary->borders(*region);
            if( !other.is_abandoned() && other.owner() != nullptr && da_us.has_war_with(other))
            {
                regions_to_attack.insert(CA_STD::make_pair(other.get_bdi_index(), &other));
            }
        });
    });

    for(CA_STD::UNORDERED_MAP<card32, CAI_REGION *>::ITERATOR itr_region = regions_to_attack.begin(), itr_region_end =
         regions_to_attack.end(); itr_region!=itr_region_end; ++itr_region)
    {        new_tasks_with_priority.push_back(
          CAI_ATTACK_REGION_TASK_SPECIFICATION::TASK_INSTANTIATION_INFO_PRIORITY_DATA ((*itr_region).second, 1.0f));
    }
}
```

Task management system

Task Generation

Resource Allocation

Resource Coordination

Task management system
Resource allocation

Tasks Mobiles

- Attack region
- Attack force
- Recruit new army
- Raid region
- Defend region
- Embed agent
- Recruit in region

Resource Mobiles

- Army 1
- Army 2
- Army 3
- Navy 1
- Agent 1

Task management system

Resource allocation

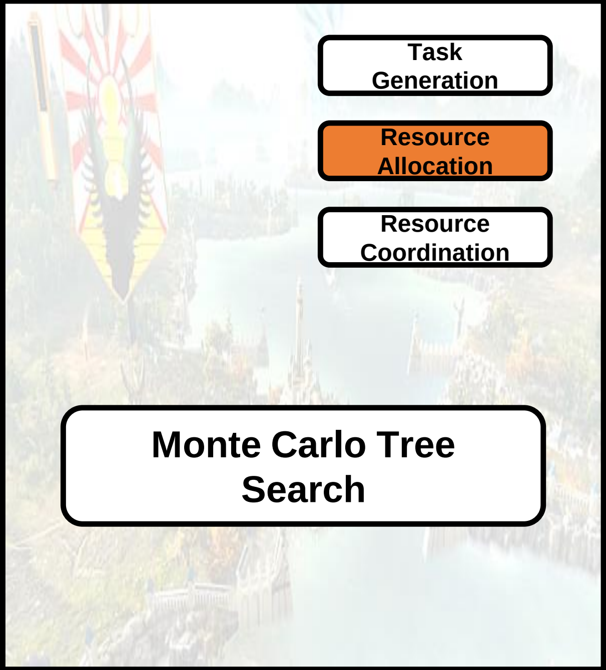
Tasks Mobiles

- Attack region
- Attack force
- Recruit new army
- Raid region
- Defend region
- Embed agent
- Recruit in region

Resource Mobiles

- Army 1
- Army 2
- Army 3
- Navy 1
- Agent 1

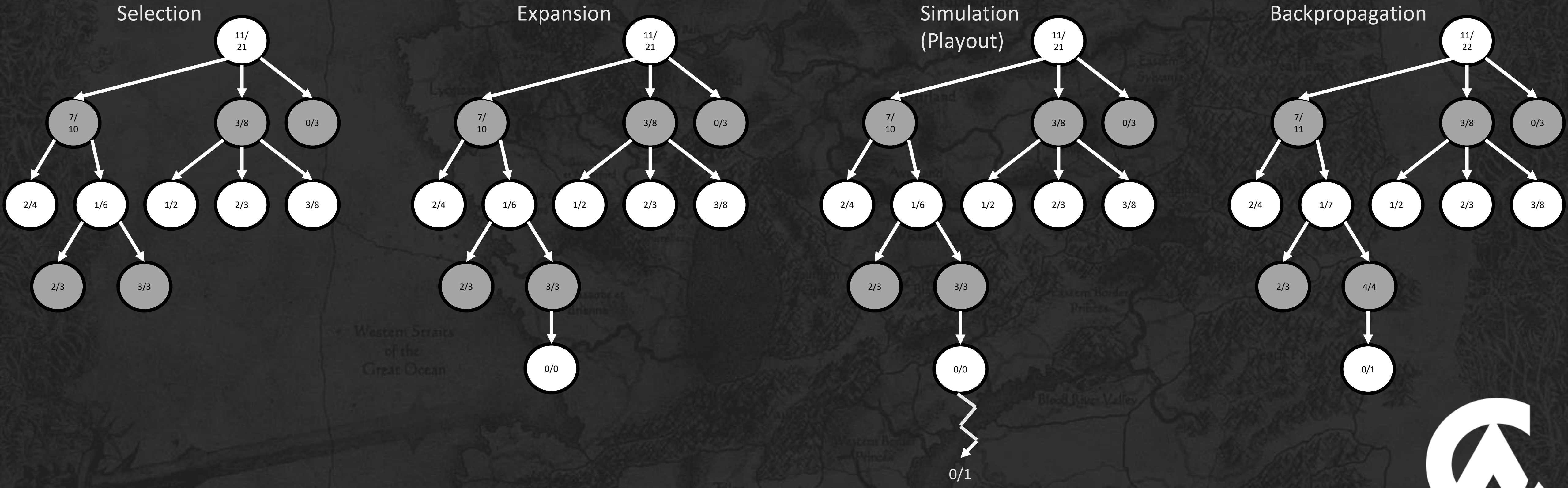
Task management system



@aphrael

Monte Carlo Tree Search

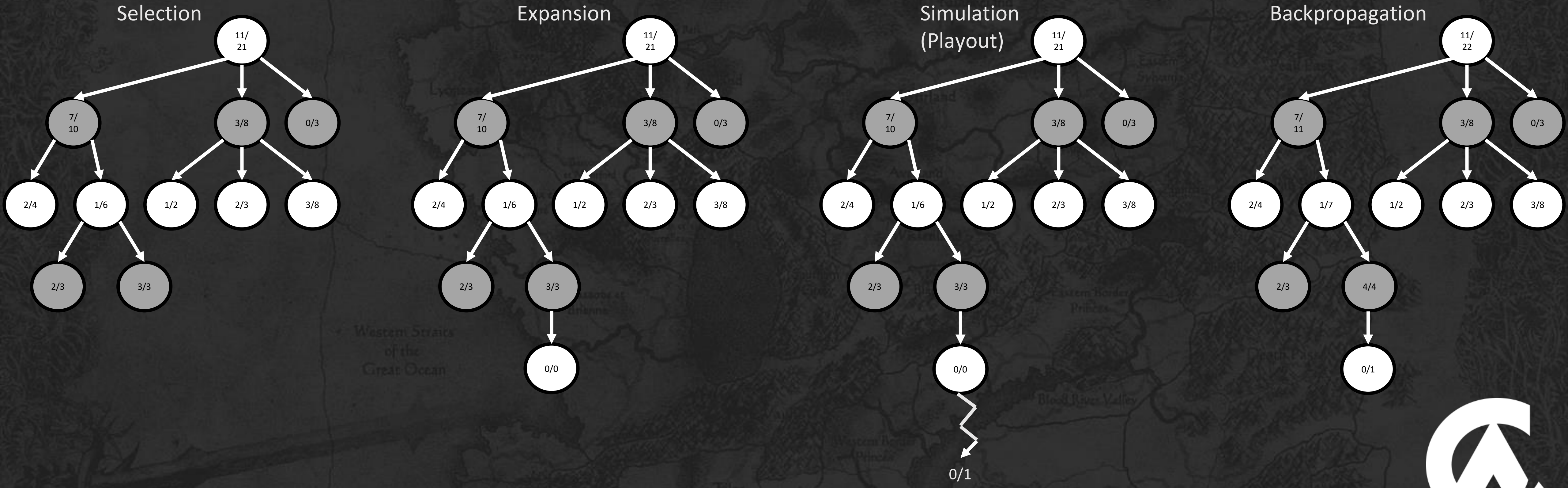
With upper confidence bound for trees, UCT



@aphrael

Monte Carlo Tree Search

With upper confidence bound for trees, UCT



@aphrael

Task management system

Task Generation

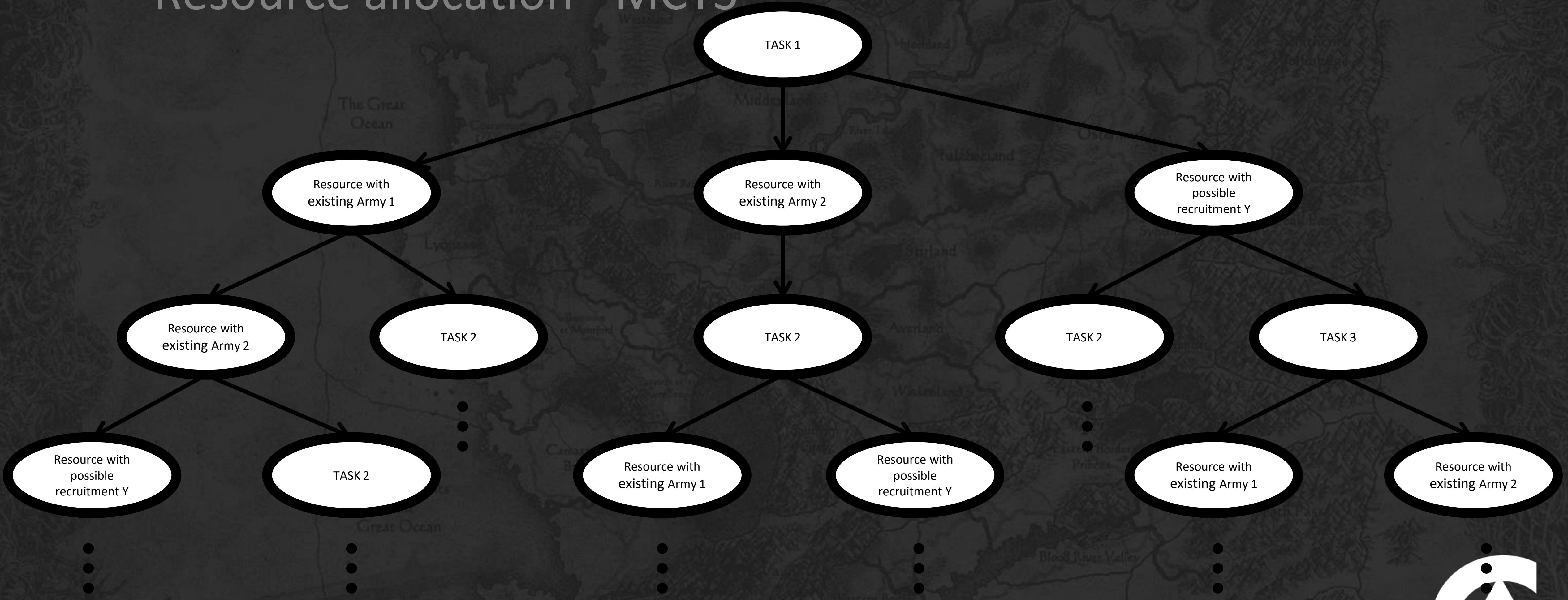
Resource Allocation

Resource Coordination

@aphrael

Task management system

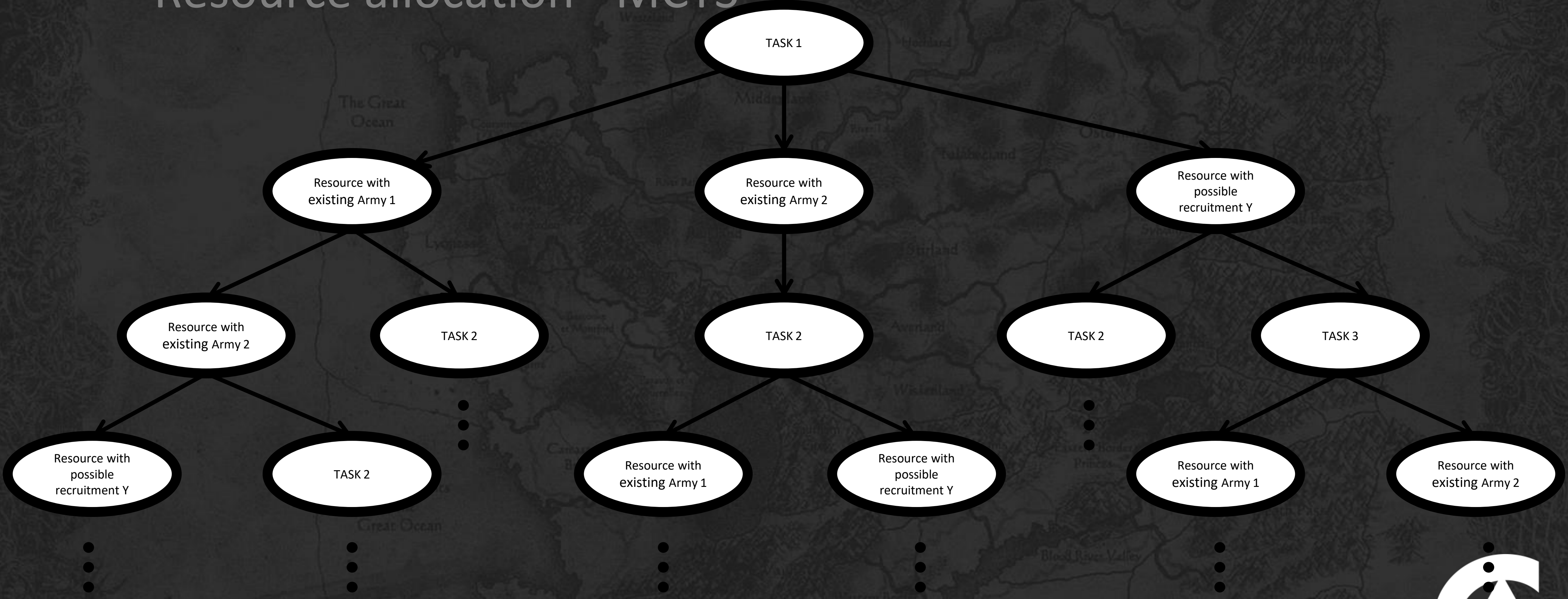
Resource allocation - MCTS


TASK 1
Resource with existing Army 1
Resource with existing Army 2
Resource with possible recruitment Y
Resource with existing Army 2
TASK 2
TASK 2
TASK 2
TASK 3
Resource with possible recruitment Y
TASK 2
Resource with existing Army 1
Resource with possible recruitment Y
Resource with existing Army 1
Resource with existing Army 2

@aphrael

Task management system

Resource allocation - MCTS


TASK 1
Resource with existing Army 1
Resource with existing Army 2
Resource with possible recruitment Y
Resource with existing Army 2
TASK 2
TASK 2
TASK 2
TASK 3
Resource with possible recruitment Y
TASK 2
Resource with existing Army 1
Resource with possible recruitment Y
Resource with existing Army 1
Resource with existing Army 2

@aphrael

Task management system

- Task Generation
- Resource Allocation
- **Resource Coordination**

@aphrael

Task management system

Resource coordination

Tasks

- Attack region
- Attack force
- Recruit new army
- Raid region
- Defend region
- Embed agent
- Recruit in region

Resource Mobiles

- Army 1
- Army 2
- Army 3
- Navy 1
- Agent 1

Embed agent 1 with Army 1

Attack Region 1 with Army 1

Recruit new Army 2 in Region 3

Defend Region 2 with Army 3 and 4

@aphrael

Task management system

Resource coordination

Tasks

- Attack region
- Attack force
- Recruit new army
- Raid region
- Defend region
- Embed agent
- Recruit in region

Resource Mobiles

- Army 1
- Army 2
- Army 3
- Navy 1
- Agent 1

Embed agent 1 with Army 1

Attack Region 1 with Army 1

Recruit new Army 2 in Region 3

Defend Region 2 with Army 3 and 4

@aphrael

Task management system

Resource coordination

Node Types

Target nodes

- Own Army
- Their Army
- Own Settlement
- Their Settlement
- Their Agents

Action nodes

- Agent Bolster Actions
- Agent Hindering Actions
- Attack Actions
- Siege/ Blockade/ Assault
- Garrison
- Move

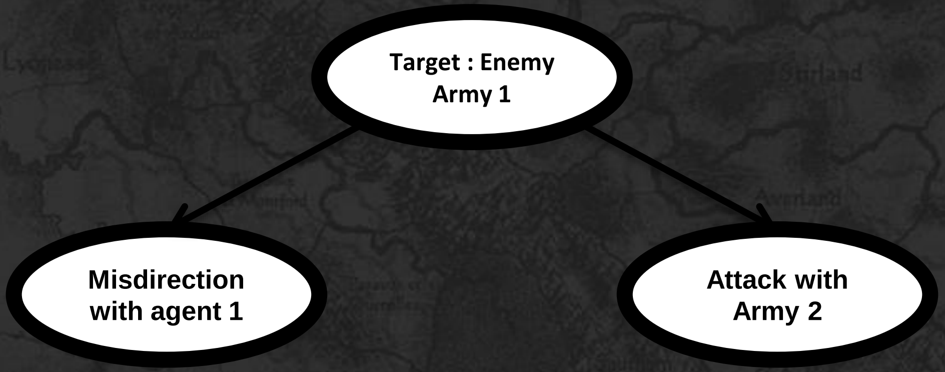
Task management system

Resource coordination

Node Types

Target nodes

- Own Army
- Their Army
- Own Settlement
- Their Settlement
- Their Agents



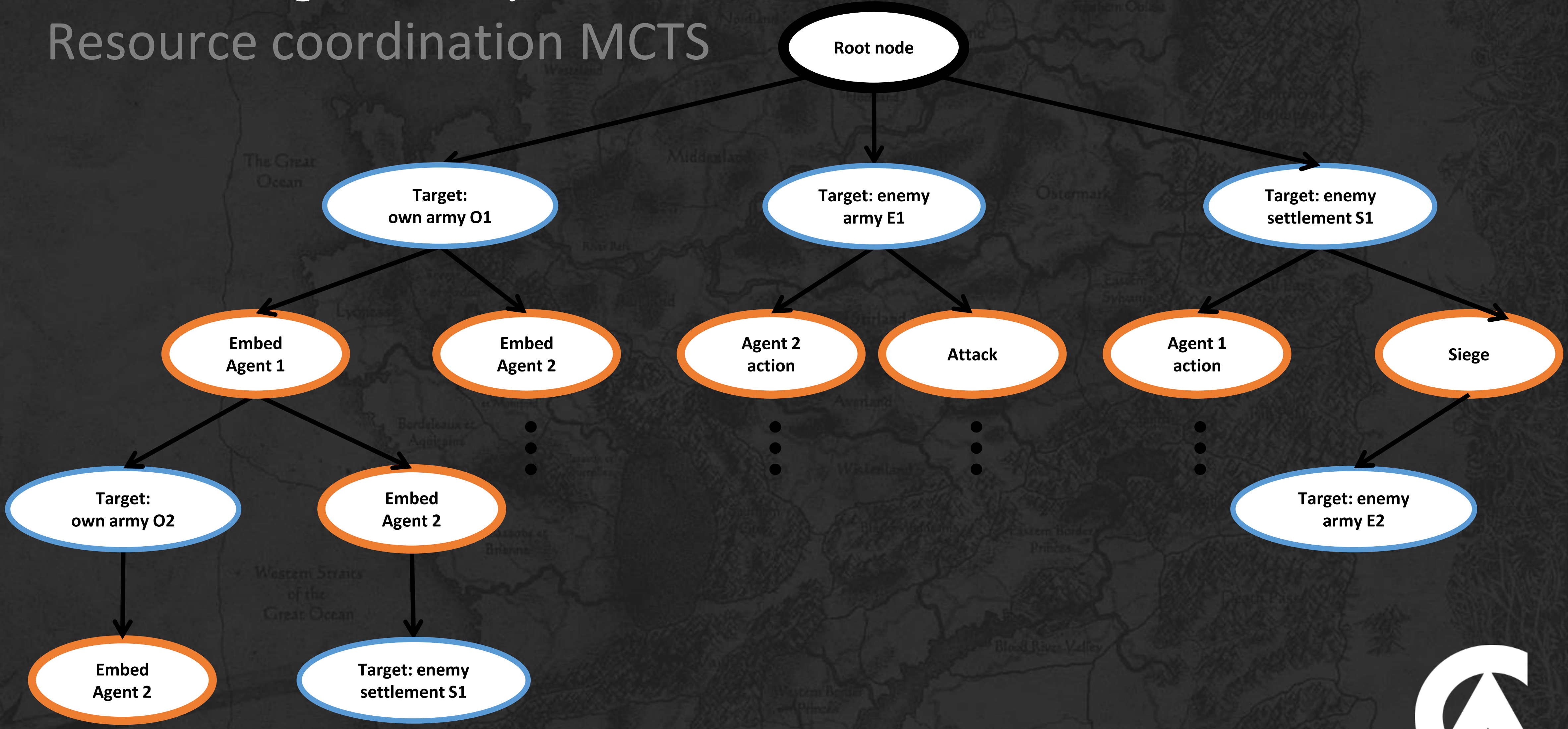
Action nodes

- Agent Bolster Actions
- Agent Hindering Actions
- Attack Actions
- Siege/ Blockade/ Assault
- Garrison
- Move

@aphrael

Task management system

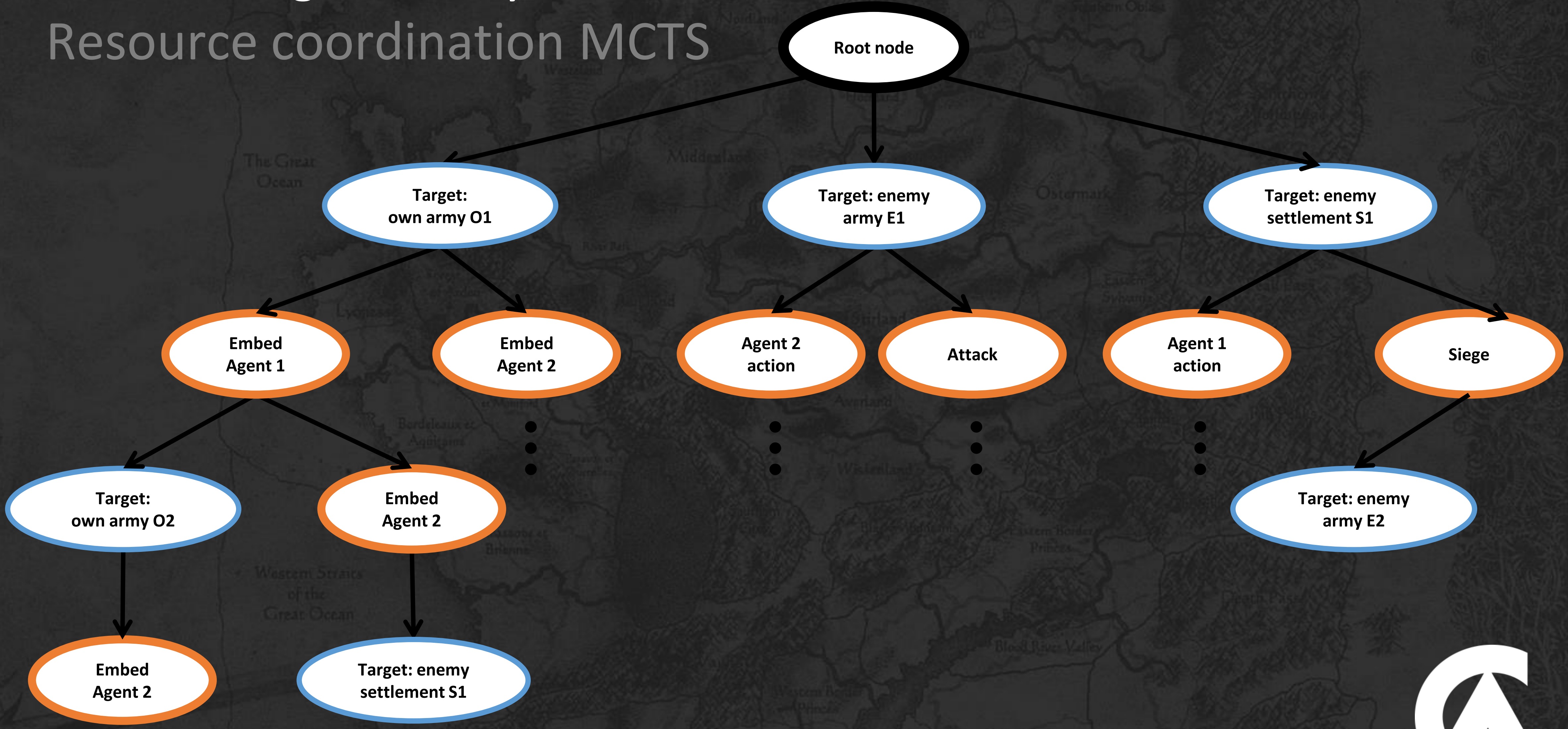
Resource coordination MCTS



@aphrael

Task management system

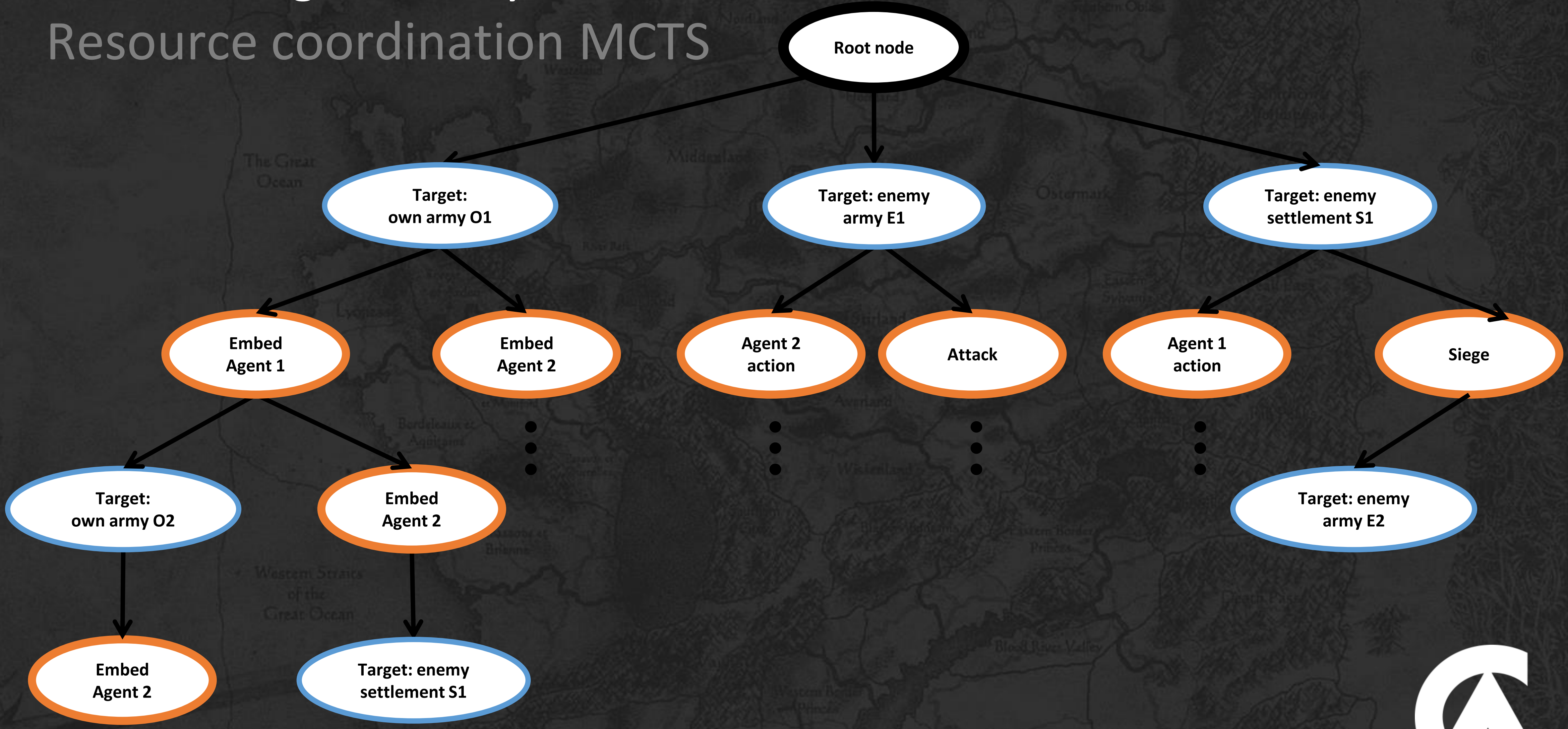
Resource coordination MCTS



@aphrael

Task management system

Resource coordination MCTS



@aphrael

Resource coordination

Optimisations

- Pruning
- Sub-phases
- Ordering
- Path caching
- Spatial partitioning

@aphrael

Resource coordination

Optimisations

Pruning

- Targets: unreachable
- Targets: attacks unsuccessful
- Targets: unsuccessful agent actions.

@aphrael

Resource coordination

Optimisations

Pruning

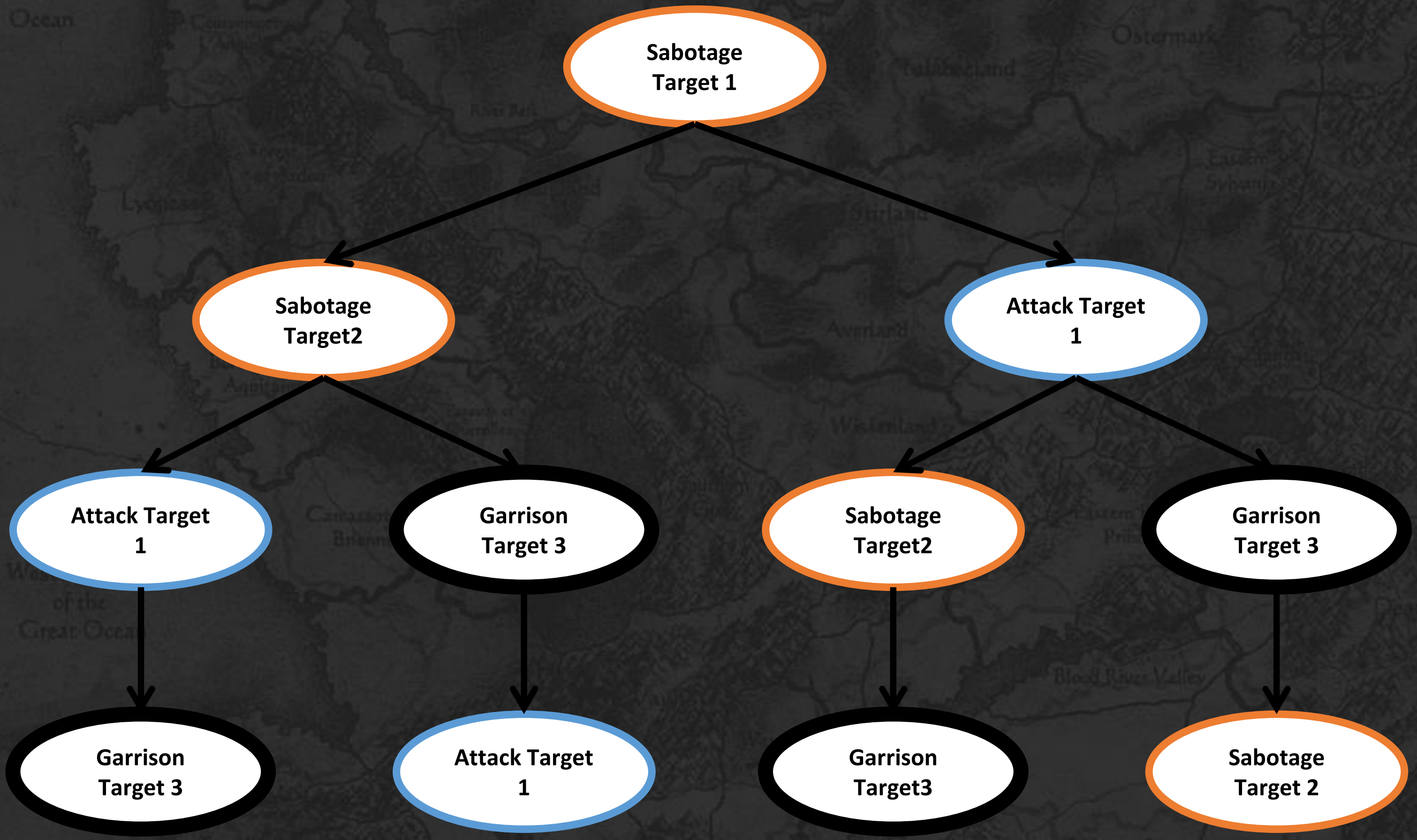
- Targets: unreachable
- Targets: attacks unsuccessful
- Targets: unsuccessful agent actions.

@aphrael

Resource coordination

Optimisations

Sub-phases

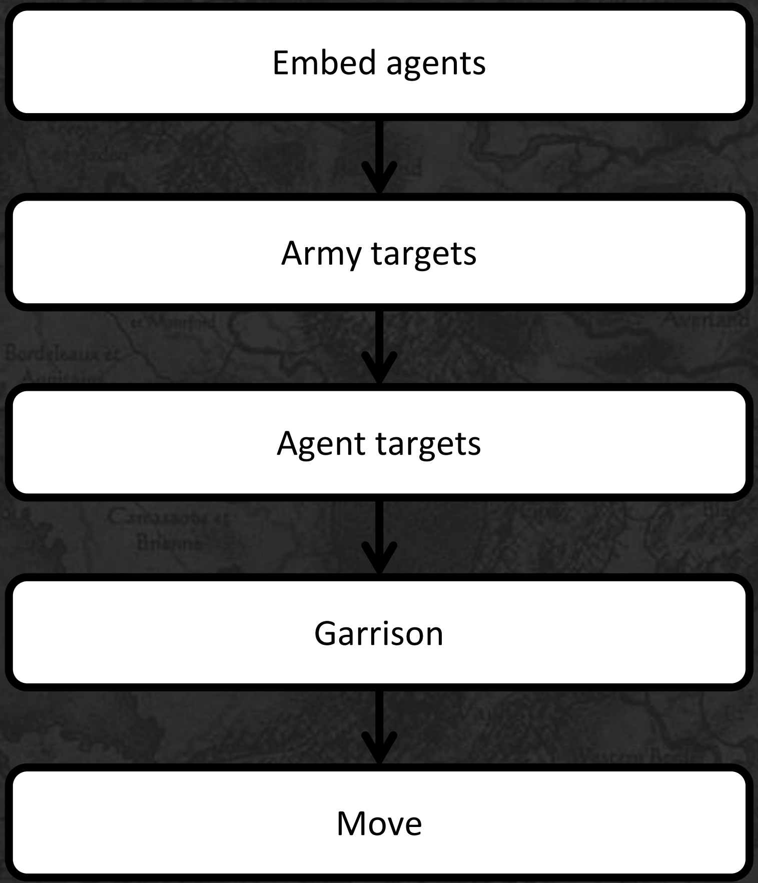


@aphrael

Resource coordination

Optimisations

Sub-phases

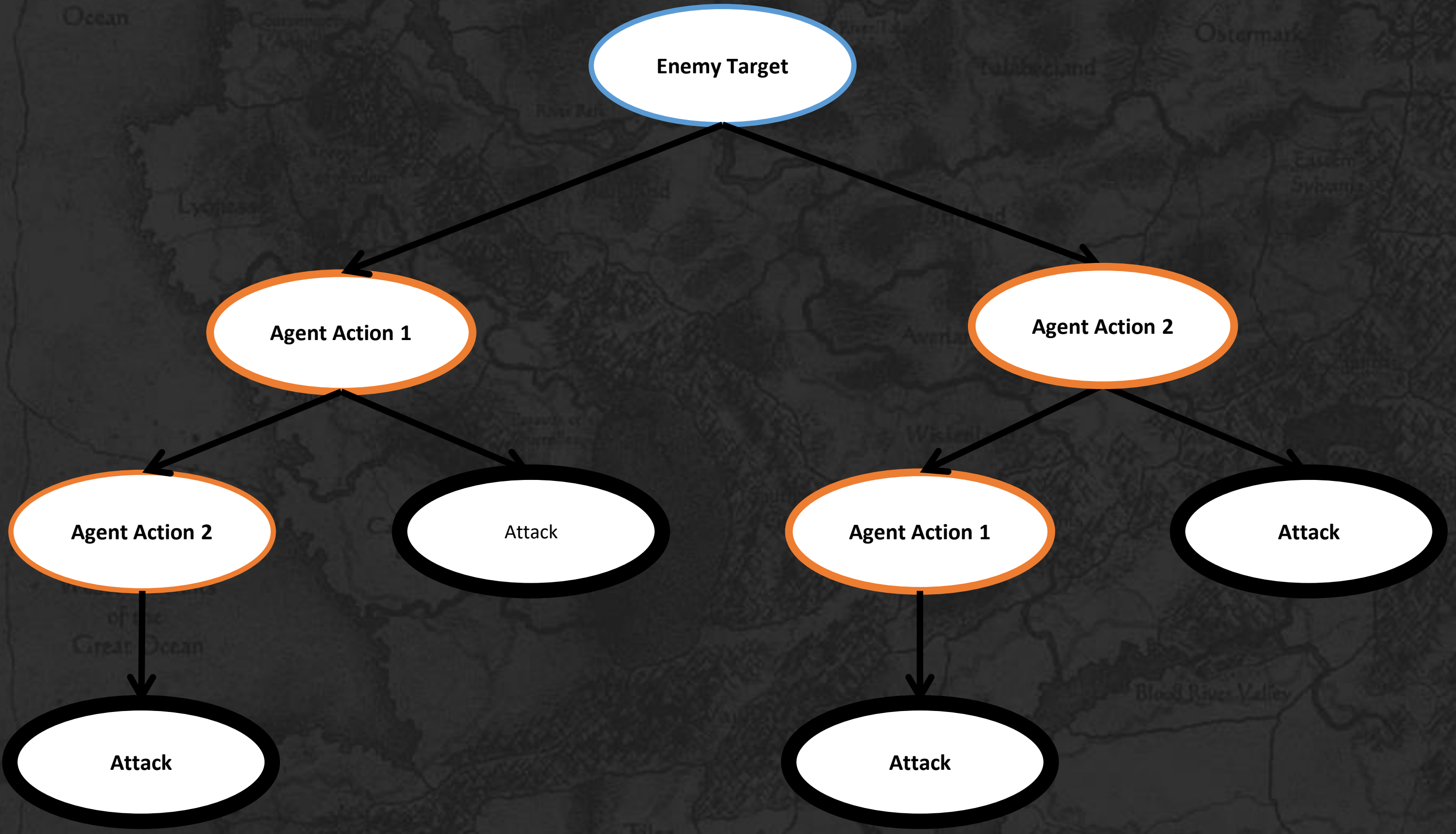


@aphrael

Resource coordination

Optimisations

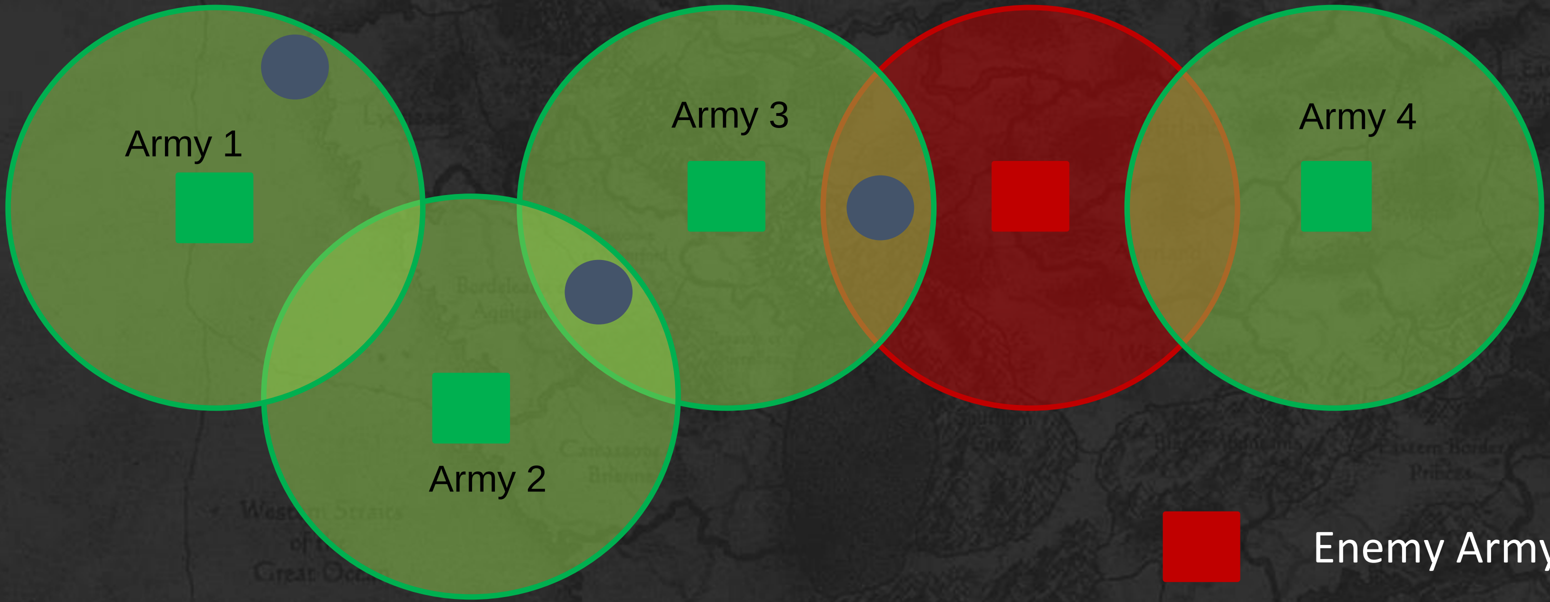
Node Ordering



Resource coordination

Optimisations

Spatial Partitioning



🟥 Enemy Army

🟩 Our Army

● Target

@aphrael

Task Management System

Summary

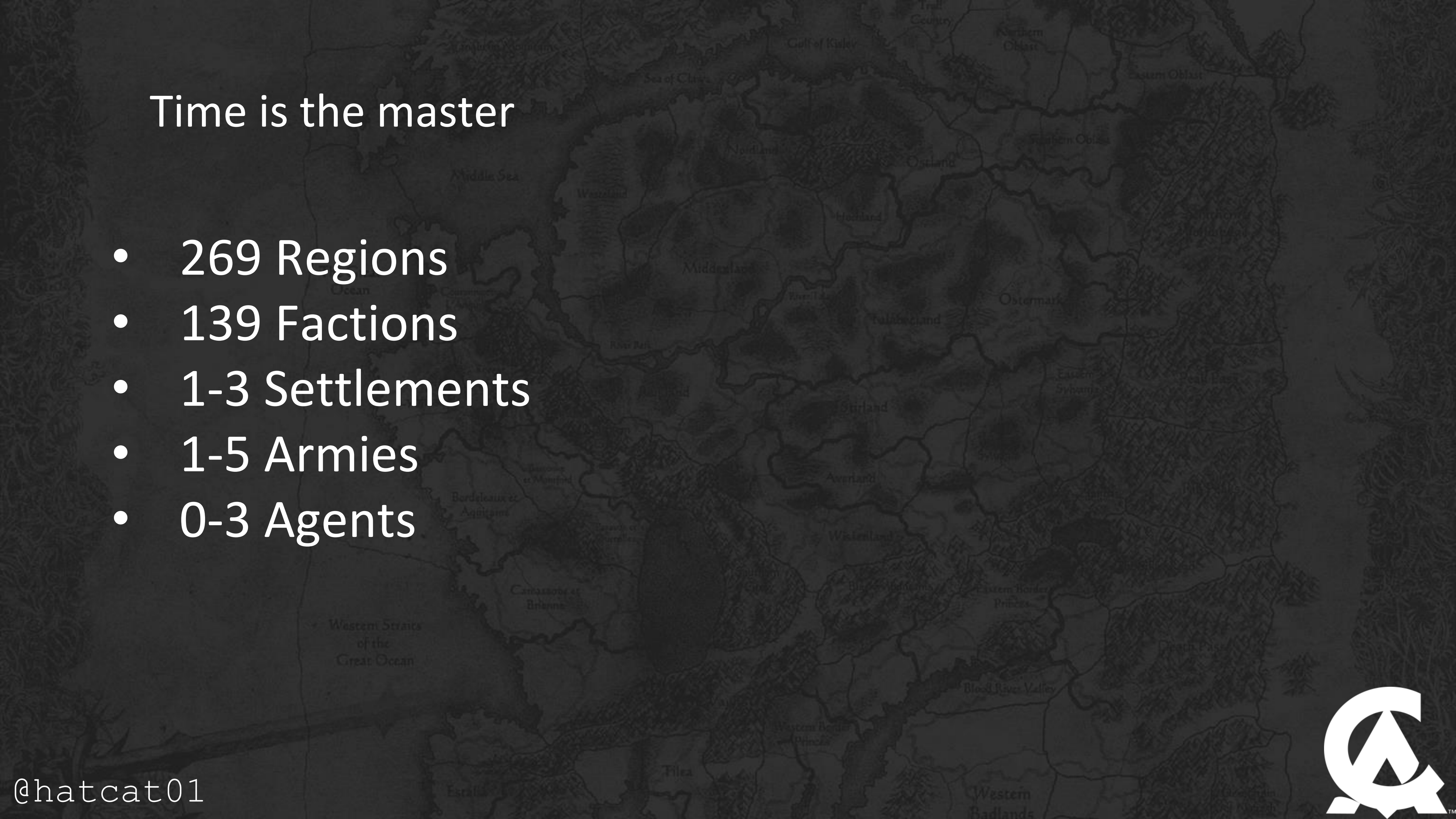
- Task Allocation and Coordination using MCTS
    - Customisable
    - Good exploitation vs exploration tradeoff
    - Anytime
- Optimisations: Pruning, Lazy evaluation, Spatial partioning

@aphrael

Overview

- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy
- Tasks and resources
- Profiling and timing

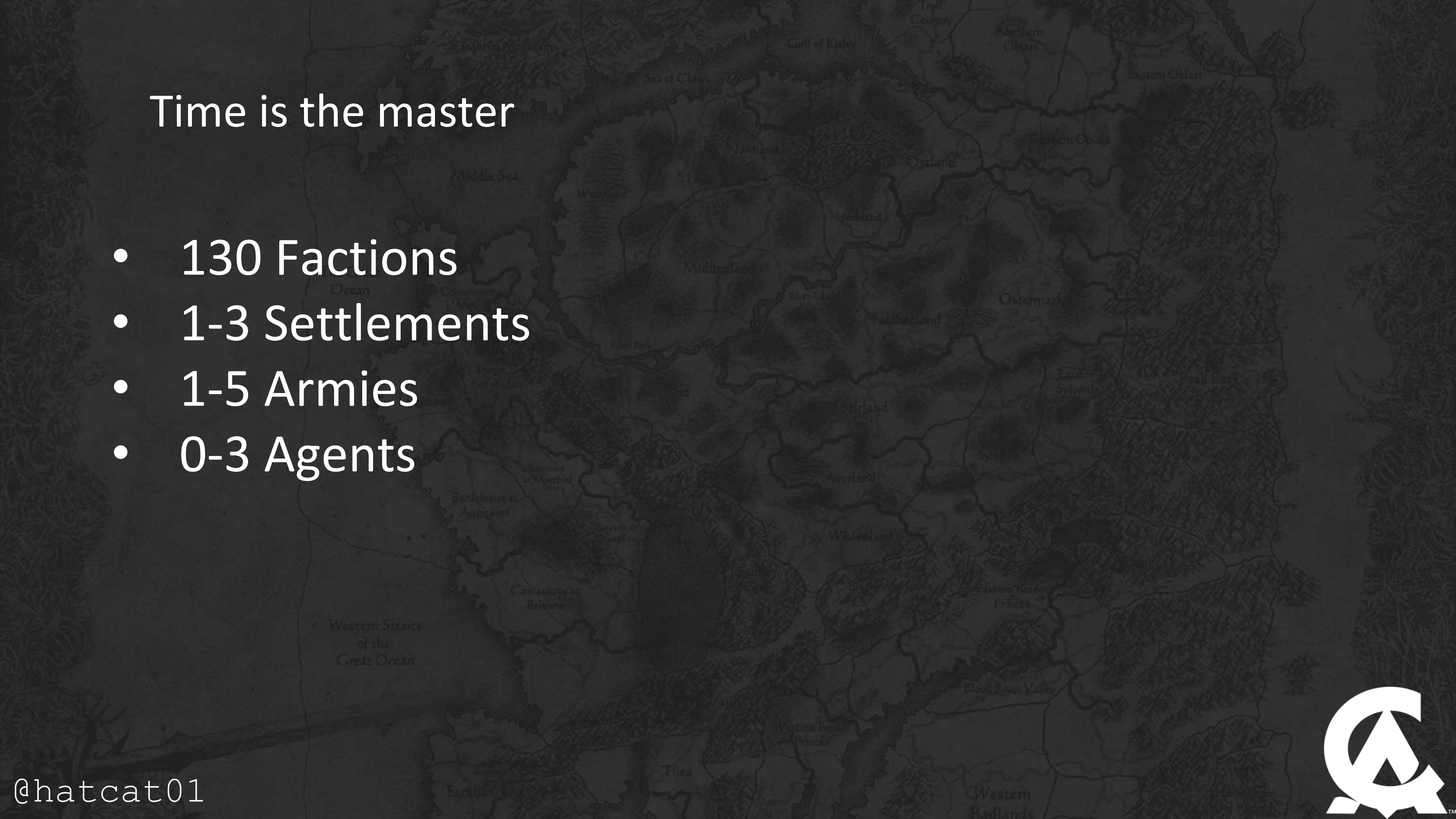
@hatcat01



Time is the master

- 269 Regions
- 139 Factions
- 1-3 Settlements
- 1-5 Armies
- 0-3 Agents

@hatcat01



Time is the master

- 130 Factions
- 1-3 Settlements
- 1-5 Armies
- 0-3 Agents

Using the cycles

- Caching
- Planners

Using the cycles

- Caching
- Long-term planners
- Quadratic => Cubic
- Profiling

Getting performance data

- Sleepy
- Telemetry collection

@hatcat01

Getting performance data

- Sleepy
- Telemetry collection
- Chrome tracing
- You can’t control what you can’t measure

Generating performance data

- “Oh, that’s weird…”
- Data series over time

@hatcat01

Generating performance data

- “Oh, that’s weird…”
- Data series over time
- Autotesting
- Step-changes

Budgets

- Varying durations
- Hardware specific

Budgets

- Varying durations
- Hardware specific
- Caps
- Continuous interference

Dividing the time

- Factions versus components
- Component versus component

@hatcat01

Dividing the time

- Factions versus components
- Component versus component
- Diplomacy
- Scalability not always availability

Doing less

- Do the minimum
- Build and repair

Doing less

- Do the minimum
- Build and repair
- Do what is observable
- Avoid what is silly

Evading responsibility

- Let the user decide
- Not everything scales (yet)
- More time for tasks

Overview

- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy

@hatcat01

Overview

- Introducing the Total War campaign
- An overview of AI systems and the world state
- A consideration of diplomacy
- Tasks and resources
- Profiling and timing
- One more thing…

Est. 1987

ONE OF EUROPE'S LONGEST-STANDING AND LARGEST STUDIOS
STATE-OF-THE-ART FACILITIES IN THE UK AND BULGARIA
MULTI-AWARD WINNING HERITAGE OF AAA TITLES

JOIN US @CAGAMES

CREATIVE-ASSEMBLY.COM

CREATIVE
ASSEMBLY™

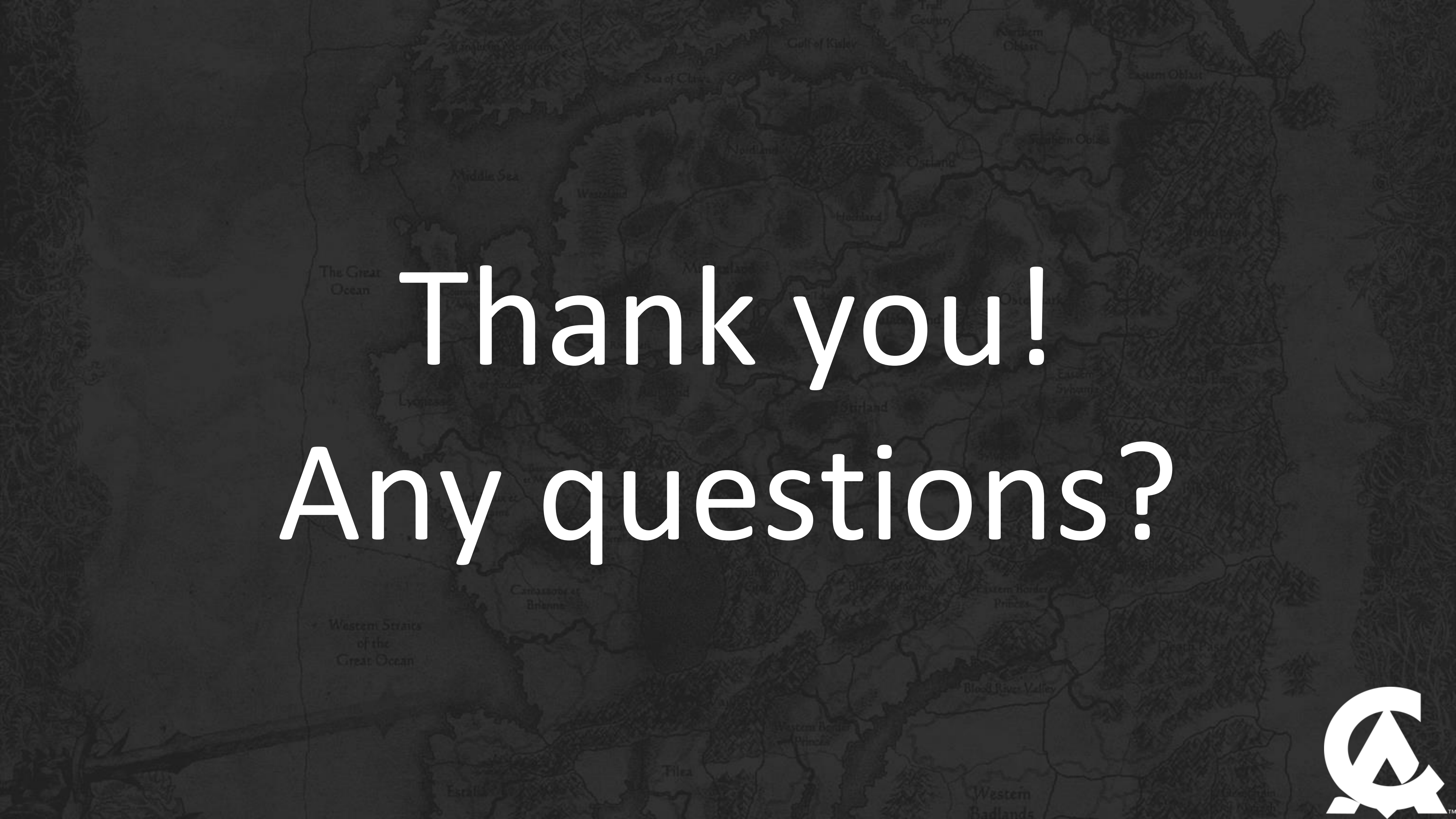
Join us?

- Tools programmer – NEW IP, console
- Engine programmer – NEW IP, console
- Backend developer – Total War Arena

@CAGames

Join us?

- Tools programmer – NEW IP, console
- Engine programmer – NEW IP, console
- Backend developer – Total War Arena
- Engine programmer – Total War Arena
- Tools programmer – Total War
- Engine programmer – Total War
- Campaign AI programmer – Total War



Thank you!
Any questions?