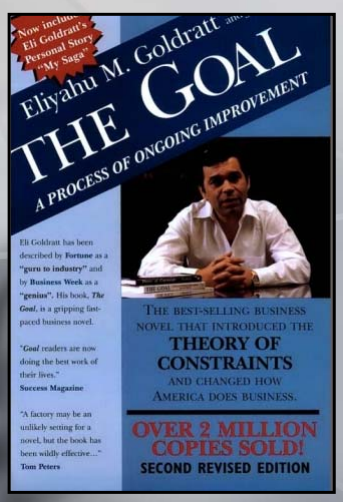
Theory Of Constraints (TOC)

Managing and Controlling the *Whole* by a *Part*

TOC meetings in South America (2007) and Europe (2005)

TOC's grounding father dr. Eli Goldratt (middle)



Video The Goal

To quote Goldratt on TOC's paradigm:

Each system is limited by a constraint which prevents it from achieving a higher performance relative to its goal.

In order to manage the performance of the system, the constraint must be identified and treated carefully.

let's examine this paradigm, let's go into questions like:

- **what is 'limited by'?**
- **what is a constraint?**
- **what is its logistic approach?**

- **How to identify a constraint?**
- **How to manage a buffer?**
- **How to improve a system?**

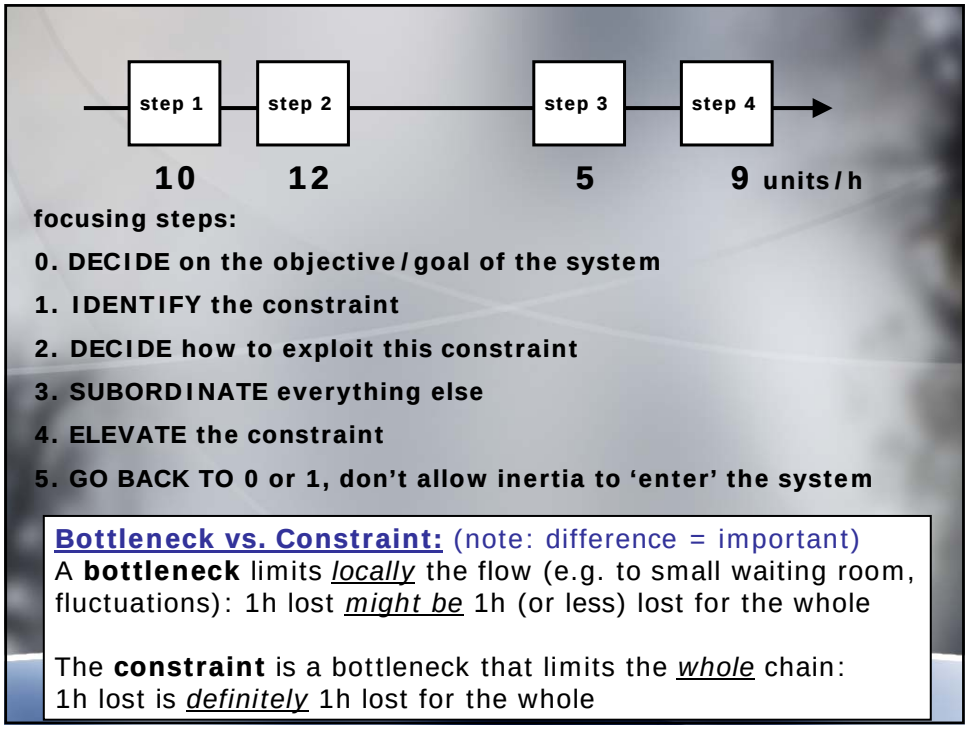
10 **12** **5** **9 units/h**

unit: product, service, but also a patient

what is the maximum output of this system per hour?

suppose you are the manager of step 2: what would you produce per hour, and why?

Paradoxical Conclusion:
the parts must perform to the max., but produced less to do the right for the system as a whole
(note: policies, rules, prevent the system of doing so)

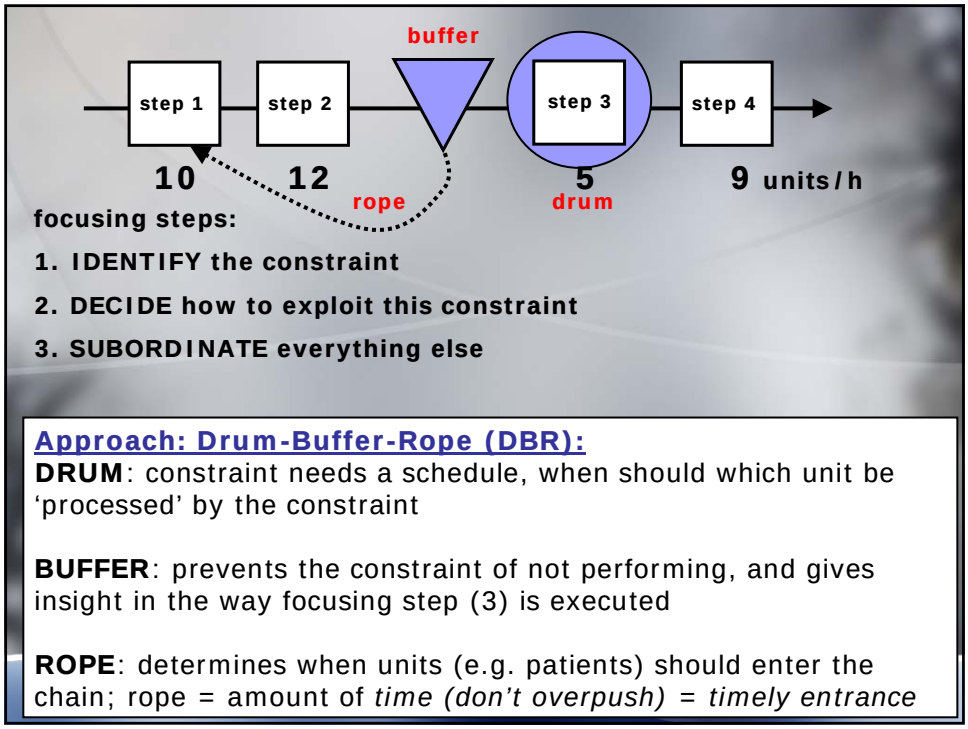


focusing steps:

0. **DECIDE on the objective/goal of the system**
1. **IDENTIFY the constraint**
2. **DECIDE how to exploit this constraint**
3. **SUBORDINATE everything else**
4. **ELEVATE the constraint**
5. **GO BACK TO 0 or 1, don't allow inertia to 'enter' the system**

Bottleneck vs. Constraint: (note: difference = important)
A **bottleneck** limits *locally* the flow (e.g. to small waiting room, fluctuations): 1h lost *might be* 1h (or less) lost for the whole

The **constraint** is a bottleneck that limits the *whole* chain: 1h lost is *definitely* 1h lost for the whole



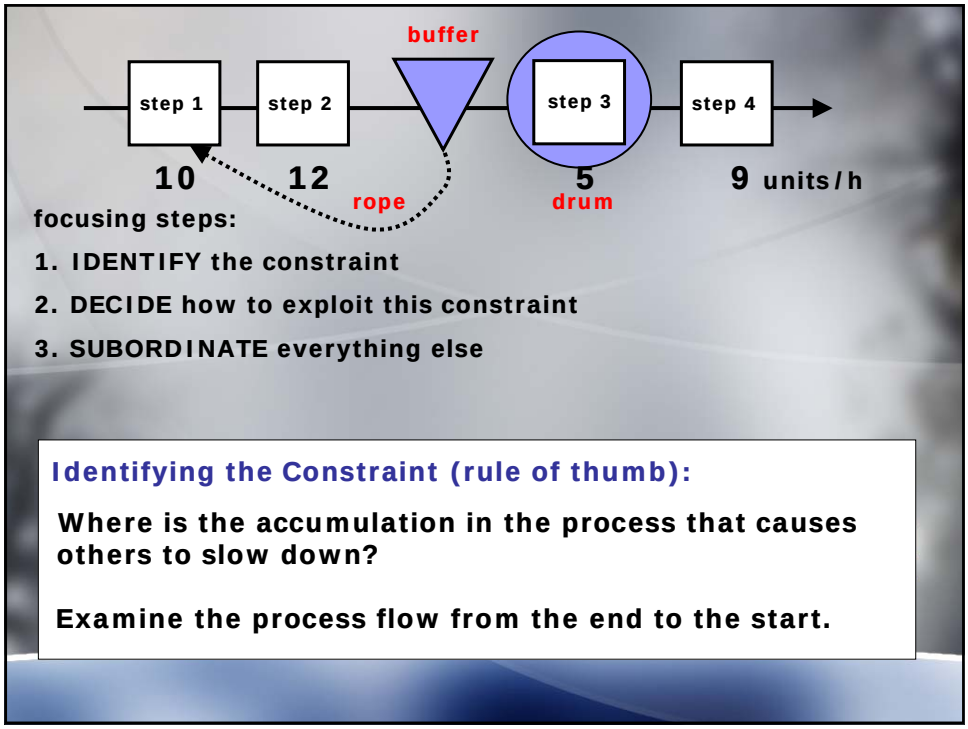
focusing steps:

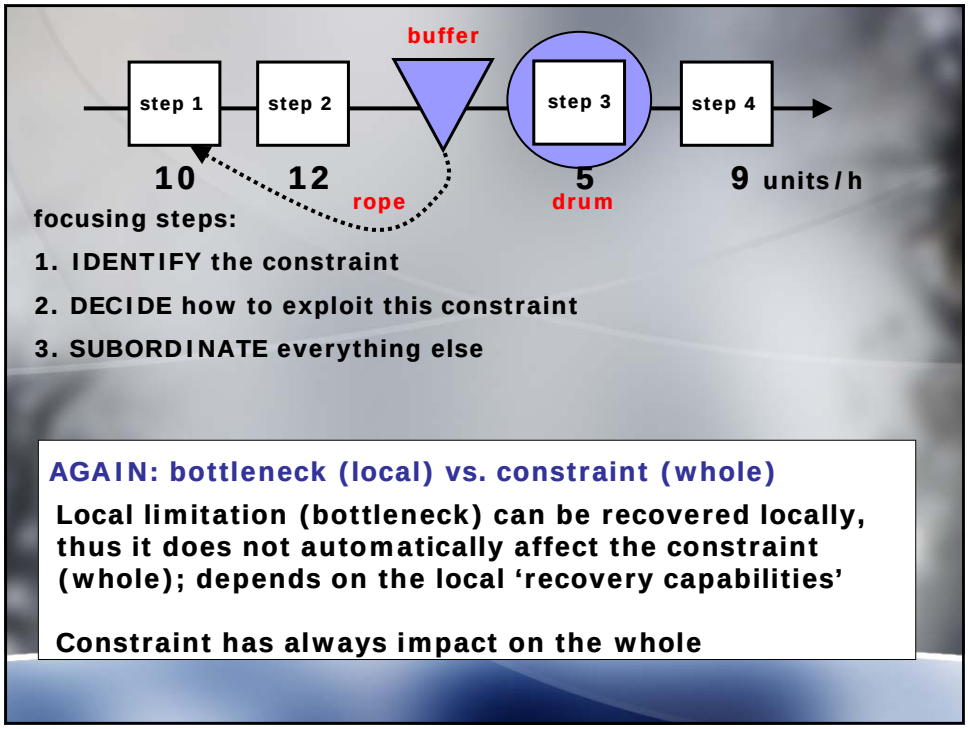
1. **IDENTIFY the constraint**
2. **DECIDE how to exploit this constraint**
3. **SUBORDINATE everything else**

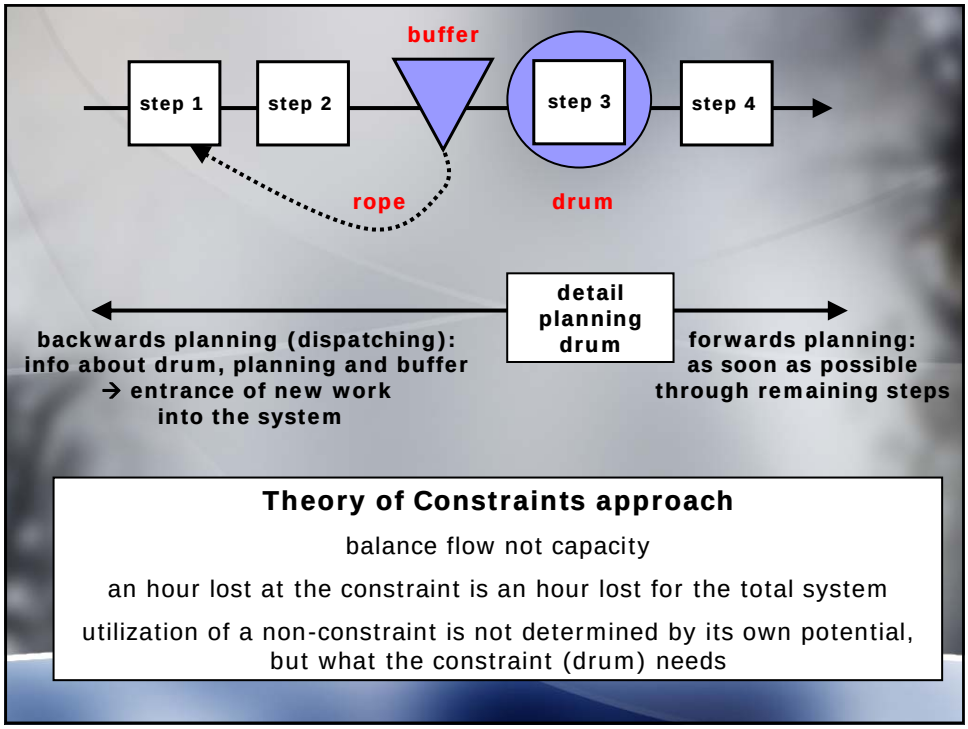
Approach: Drum-Buffer-Rope (DBR):
DRUM: constraint needs a schedule, when should which unit be 'processed' by the constraint

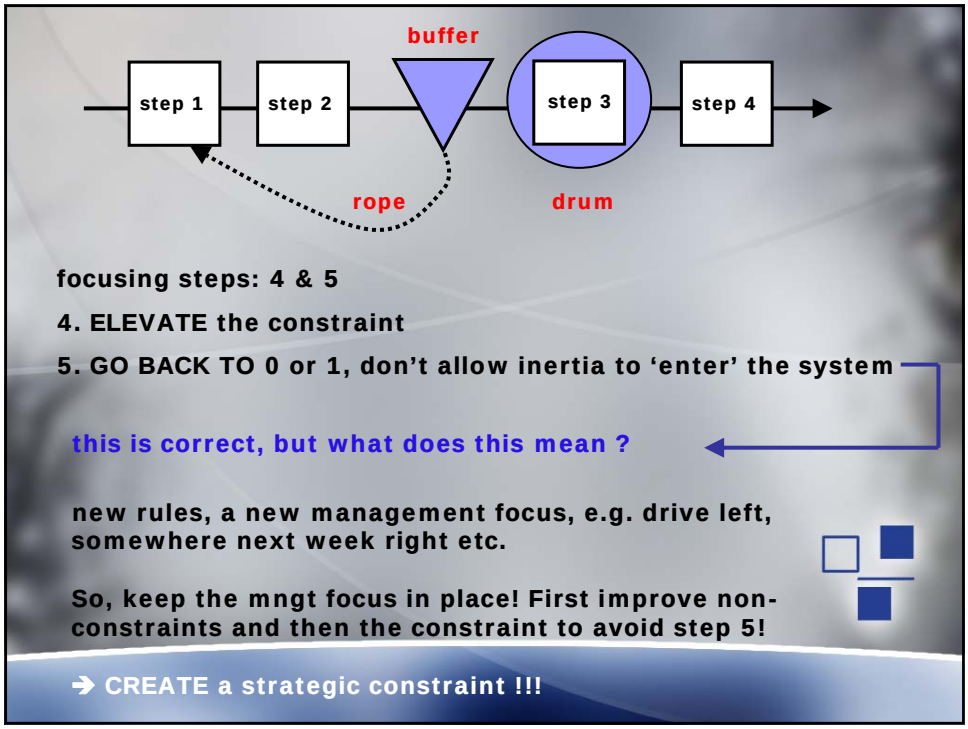
BUFFER: prevents the constraint of not performing, and gives insight in the way focusing step (3) is executed

ROPE: determines when units (e.g. patients) should enter the chain; rope = amount of *time (don't overpush) = timely entrance*


buffer
step 1
step 2
step 3
step 4
10
12
5
9 units/h
rope
drum
focusing steps:
1. IDENTIFY the constraint
2. DECIDE how to exploit this constraint
3. SUBORDINATE everything else
Identifying the Constraint (rule of thumb):
Where is the accumulation in the process that causes others to slow down?
Examine the process flow from the end to the start.

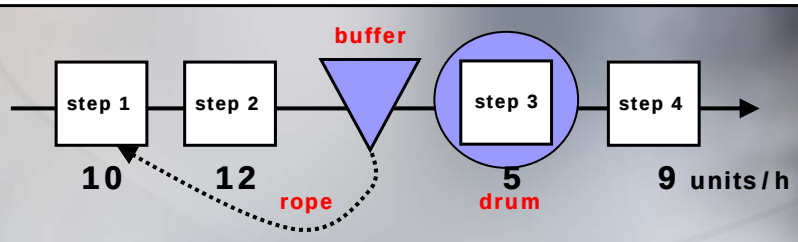

buffer
step 1
step 2
step 3
step 4
10
12
5
9 units/h
rope
drum
focusing steps:
1. IDENTIFY the constraint
2. DECIDE how to exploit this constraint
3. SUBORDINATE everything else
AGAIN: bottleneck (local) vs. constraint (whole)
Local limitation (bottleneck) can be recovered locally, thus it does not automatically affect the constraint (whole); depends on the local 'recovery capabilities'
Constraint has always impact on the whole


buffer
step 1
step 2
step 3
step 4
rope
drum
detail planning drum
backwards planning (dispatching): info about drum, planning and buffer → entrance of new work into the system
forwards planning: as soon as possible through remaining steps
Theory of Constraints approach
balance flow not capacity
an hour lost at the constraint is an hour lost for the total system
utilization of a non-constraint is not determined by its own potential, but what the constraint (drum) needs


buffer
step 1
step 2
step 3
step 4
rope
drum
focusing steps: 4 & 5
4. ELEVATE the constraint
5. GO BACK TO 0 or 1, don't allow inertia to 'enter' the system
this is correct, but what does this mean ?
new rules, a new management focus, e.g. drive left, somewhere next week right etc.
So, keep the mngt focus in place! First improve non-constraints and then the constraint to avoid step 5!
→ CREATE a strategic constraint !!!

How to Manage the Buffer (BM)?

buffer size, and in- and outflow of buffer



Buffer size? **Depends on the drum and the previous steps**

Drum needs a new unit each 12 min. (60/5), and from start to drum takes 11 min. (60/(10+12))

Thus, the rope (release of order) is at least 11 min. *before* the drum needs it. In theory, the buffer size is max. 1 unit.

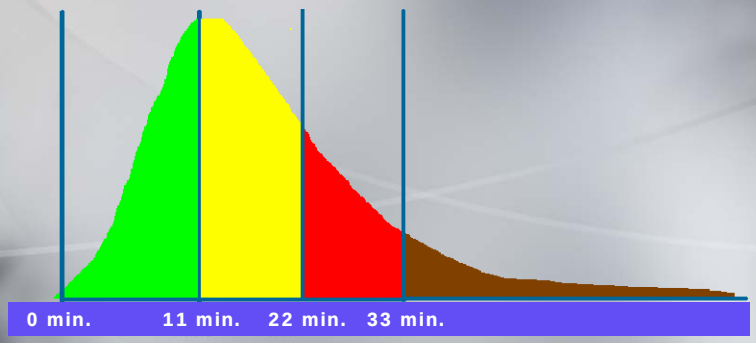
But to protect us against Murphy, we can size the buffer to e.g. 3 units; rope is then 33 min. (3*11 min.)

If a hick-up occurs in the first steps, you have 22 min. extra before anything happens in the system as a whole; the buffer will be consumed.

Buffer can be replenished quickly, because the first steps are able to produce faster that the constraint can consumed the buffer.

Buffer Management

Split the 33 min. in 3 zones: green, yellow, red.



Actively manage units in the yellow zone to prevent them from coming in the red zone

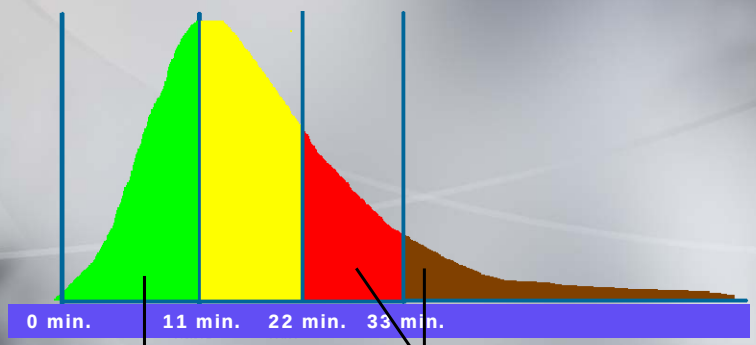
Units in the red/brown zone need to be expedited for the remaining steps (here: 1 and 2), else they are to late → drum has to stop

Expedited: change transfer rate between steps

How to Improve the System?

focusing attention based on buffer signals

Signalling



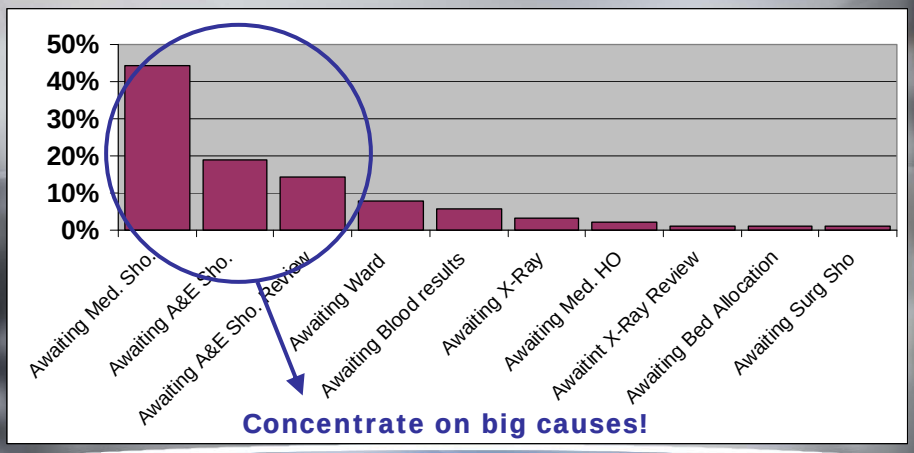
If units are mostly here then the buffer runs 'over' structurally

'No' action

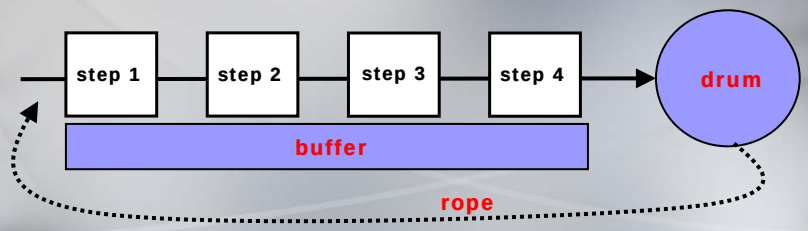
If units are mostly here then the buffer runs 'dry' structurally

Action: elimination of causes, or buffer / rope enlargement

Via Buffer Analysis = Pareto Analysis



Working with Due-Dates: e.g. service must be delivered within 4 hours



Where is the drum?

So, not a physical constraint but a due-date constraint

What (where) is the buffer and rope?

Rope is the 'doorman' which controls entrance level

Would you perform a heart surgery after you attend a lecture on this subject ?

Be aware that this also applies for TOC (DBR/BM), it requires profound knowledge !